
# FEASIBILITY OF VERY LARGE SPARSE APERTURE DEPLOYABLE ANTENNAS

THESIS

Jason C. Heller, Captain, USAF

AFIT-ENY-14-M-24

**DEPARTMENT OF THE AIR FORCE**
**AIR UNIVERSITY**

***AIR FORCE INSTITUTE OF TECHNOLOGY***

**Wright-Patterson Air Force Base, Ohio**

AFIT-ENY-14-M-24

# FEASIBILITY OF VERY LARGE SPARSE APERTURE DEPLOYABLE ANTENNAS

THESIS

Presented to the Faculty

Department of Aeronautical and Astronautical Engineering

Graduate School of Engineering and Management

Air Force Institute of Technology

Air University

Air Education and Training Command

In Partial Fulfillment of the Requirements for the

Degree of Master of Science in Astronautical Engineering

Jason C. Heller, BS

Captain, USAF

March 2014

AFIT-ENY-14-M-24

# FEASIBILITY OF VERY LARGE SPARSE APERTURE DEPLOYABLE ANTENNAS

Jason C. Heller, B.S., Aerospace Engineering
Captain, USAF

Approved:

| | |
|---|---|
| _____//signed//_____ | _13 MAR 14__ |
| Dr. Alan L. Jennings (Chairman) | Date |
| _____//signed//_____ | _13 MAR 14__ |
| Dr. Jonathan T. Black (Member) | Date |
| _____//signed//_____ | _13 MAR 14__ |
| Dr. Richard G. Cobb (Member) | Date |

## Abstract

The objective of this research is to explore the technical soundness of a very large, cross-shaped, parabolic, sparse aperture antenna extending 75 m from the bus. Specifically, describing the environment of the satellite, the effect of fabrication error on the structure and the remaining error budget for the system. The methodology involves creation of an ideal truss structure, to which all others are compared. A uniform distribution of proportional errors up to 1e-5 is introduced into the truss members' lengths and the models are subjected to a static Finite Element Analysis. A solution for the surface normal error is addressed using Lagrange multipliers. The goal is to hold the surface normal error for the entire satellite below a root mean square of 15 mm. The analysis yields a surface error of less than 1.53 mm, well within requirements. Despite the enormous size of the antenna reflector, and tight diameter/surface error ratio of 10,000 required for L-band communication, the system seems feasible. The values achieved for truss induced surface errors are in line with established techniques for analyzing full aperture, and strip, mesh antennas. With the mesh reflector and truss largely defined, nearly half of the 15 mm error budget remains.

*To the Friends and Family who were there when I came up for air.*

**Acknowledgments**

I would like to express my sincere appreciation to my research advisor, Dr. Alan Jennings, for his support throughout the course of this thesis effort. Without his guidance the work herein would not have been possible. I would also like to thank Dr. Jonathan Black and Dr. Richard Cobb for their time and expertise in helping me to complete this research.

Jason C. Heller

## Table of Contents

Page

Abstract ..... iv

Acknowledgments ..... vi

Table of Contents ..... vii

List of Figures ..... ix

List of Tables ..... x

List of Equations ..... xi

List of Acronyms ..... xiii

List of Symbols ..... xiv

I. Introduction ..... 1

1.1 Background ..... 1
*1.1.1 Very Large Antennas ..... 1*
*1.1.2 Sparse Aperture Antennas ..... 3*
*1.1.3 RMS Surface Error ..... 5*
1.2 Research Objectives ..... 6
1.3 Materials & Equipment ..... 7
1.4 Scope ..... 7
1.5 Overview ..... 8

II. Background & Literature Review ..... 9

2.1 Geometry of Cross-Shaped Flat Reflector ..... 9
2.2 Surface Errors in a Large Space-Based Antenna ..... 11
2.3 Summary ..... 16

III. Operational Considerations ..... 17

3.1 Perturbation Drifts ..... 18
3.2 Disturbance Torques ..... 19
3.3 Summary ..... 22

IV. Methodology of Error Determination ..... 23

4.1 Analysis of Geometry ..... 23
4.2 Inducing Errors in Truss Members ..... 27

4.3 Static FEA ..... 29
4.4 Solution for Error Normal ..... 31
4.5 Summary ..... 35

V. Analysis and Results ..... 36

5.1 Analysis of Errors in the Truss Structure ..... 36
5.2 Analysis of Errors Due to Mesh Tessellation ..... 40
5.3 Comparison of Cross-Shaped Reflector to References ..... 43
5.4 Effect of Errors in Individual Truss Members ..... 46
5.5 Summary ..... 51

VI. Conclusions and Recommendations ..... 52

6.1 Conclusions of Research ..... 53
6.2 Significance of Research ..... 53
6.3 Recommendations for Future Research ..... 54

Appendix A – MATLAB® Code ..... 56

Bibliography ..... 85

Vita ..... 87

## List of Figures

Page

Figure 1: Flat Reflector Box Truss ..... 4

Figure 2: Comparison of Interference in EM Wave ..... 5

Figure 3: Acceptable Phase Offset ..... 6

Figure 4: Parabolic Reflector Box Truss ..... 10

Figure 5: Ideal 2D Parabolic Reflector ..... 24

Figure 6: Axial Curvature in Mesh ..... 25

Figure 7: 3D Reflector Box Truss ..... 27

Figure 8: Box Truss Bay Face ..... 29

Figure 9: Line Normal to Surface Through Actual Point ..... 34

Figure 10: Ideal Truss Structure ..... 37

Figure 11: Surface Normal Errors in Ideal Truss ..... 37

Figure 12: Corrected & Uncorrected RMS Errors with Errors in All Members ..... 39

Figure 13: Corrected RMS Surface Errors with Errors in All Members ..... 40

Figure 14: Rectangular Mesh Panel ..... 41

Figure 15: Cross-Section View of Mesh Panels and Paraboloid ..... 42

Figure 16: Surface Normal Errors by Location ..... 43

Figure 17: $\varepsilon_{RMS}$ vs. Number of Bays in a Truss ..... 44

Figure 18: Corrected RMS Surface Errors by Member Type ..... 47

Figure 19: Batten Error Directions ..... 49

Figure 20: RMS Error with Greater Variation ..... 51

## List of Tables

Page

Table 1: Cross-Shaped Sparse Aperture Truss Members ..... 9

Table 2: Mesh Tessellation Pattern ..... 11

Table 3: Diameter to RMS Surface Error ..... 12

Table 4: Station Keeping ΔV Requirements ..... 19

Table 5: Max Environmental Torques ..... 22

Table 6: Carbon Fiber Properties ..... 30

Table 7: Truss Member Properties ..... 30

Table 8: Analysis Parameters for Errors in All Truss Members ..... 38

Table 9: Results of Errors in Truss Members ..... 50

Table 10: Antenna Surface Normal Error Budget ..... 53

## List of Equations


Page

Equation 1: Bit Energy Ratio ..... 2

Equation 2: Energy per Bit ..... 2

Equation 3: Link Equation ..... 2

Equation 4: Antenna Gain ..... 3

Equation 5: Antenna Collection Area ..... 4

Equation 6: Mesh Facet to Surface $\varepsilon_{RMS}$ ..... 13

Equation 7: Truss Accuracy to Surface $\varepsilon_{RMS}$ ..... 14

Equation 8: Truss Diameter RMS Error ..... 14

Equation 9: Rim RMS Error ..... 14

Equation 10: Surface $\varepsilon_{RMS}$ to Surface Deviation ..... 15

Equation 11: Radiation Beam Diameter on Earth ..... 17

Equation 12: $\Delta V$ to Combat Lunar Drift ..... 18

Equation 13: $\Delta V$ to Combat Solar Drift ..... 18

Equation 14: $\Delta V$ to Combat J22 ..... 19

Equation 15: Torque due to Solar Radiation Pressure ..... 20

Equation 16: Torque due to Gravity Gradient ..... 21

Equation 17: Torque due to Magnetic Moment ..... 21

Equation 18: Parabolic Equation ..... 23

Equation 19: Longeron Length Parabolic Constraint ..... 25

Equation 20: Length and Location of Bottom Longerons ..... 26

Equation 21: Truss Errors ..... 28

Equation 22: Length of Diagonal Cables ..... 28

Equation 23: Cable Model CTE ..... 31

Equation 24: Lagrangian for Surface Normal Point ..... 32

Equation 25: Partial Lagrangians ..... 32

Equation 26: Relation Between Ideal and Actual Points ..... 33

Equation 27: Truss Accuracy to Surface $\varepsilon_{RMS}$ ..... 45

Equation 28: Mesh Facet to Surface $\varepsilon_{RMS}$ ..... 45

## List of Acronyms

| Acronym | Definition |
|---|---|
| TDRS | Tracking and Data Relay Satellite |
| BER | Bit Error Rate |
| FEA | Finite Element Analysis |
| MoI | Moments of Inertia |

## List of Symbols

| Symbol | Definition |
|---|---|
| $\pi$ | Mathematical constant 3.1415… |
| $\lambda$ | Wavelength of antenna carrier frequency |
| $\varepsilon$ | Root-mean-square error |
| $D$ | Diameter |
| $r$ | Radius |
| $l$ | Length |
| $c$ | Speed of light |
| $p$ | Distance to focus from parabola apex |
| $\sigma$ | Standard deviation |

# FEASIBILITY OF VERY LARGE SPARSE APERTURE DEPLOYABLE ANTENNAS

## I. Introduction

This research studied the feasibility of utilizing very large reflector antennas as communication links at geostationary orbit. More specifically, the design for a flat x-shaped sparse reflector from an earlier study was extended to a parabolic dish. The space-based system was analyzed to generate typical spacecraft bus structure values and an operating environment for the satellite. Then, the effects of geometric error in the structure were quantified for comparison to an ideal parabolic reflector. The goal of this research is to demonstrate the possibility of using very large sparse aperture reflectors in future communication systems.

### 1.1 Background

#### *1.1.1 Very Large Antennas*

This research is motivated by the ever increasing bandwidth requirement of communication systems. In 1995 the Tracking and Data Relay Satellite (TDRS) 7 had a downlink data rate of 108 kbps [1]. With the launch of TDRS-H in late 2000 the two 15 foot antennas could achieve a data rate of 800 Mbps [2]. Digital communication systems typically require a specific Bit Error Rate (BER) to function properly, on the order of $10^{-11}$ for data links [1]. The BER is proportional to the bit-energy to noise-spectral density ratio, Equation 1 [1]:

$$BER \approx \frac{1}{m} erfc\left(\sqrt{\frac{E_b}{N_0}} \sin\left(\frac{\pi}{M}\right)\right) \quad \textbf{(1)}$$

where:
$m$ = Number of bits per symbol
$erfc$ = Complementary error function
$E_b$ = Energy per bit
$N_0$ = Noise spectral density,
$M$ = States of the phase-shift keying method

Everything in Equation 1 is generally independent of antenna design, related to either the ambient temperature or signal modulation. The main design parameter that can be improved by antenna design is to increase the energy per bit, $E_b$, defined by Equation 2 [3].

$$E_b = P_R * T_b \quad \textbf{(2)}$$

where:
$P_R$ = Power received
$T_b$ = Bit period

Here we can increase the power received at either transmit or receive through the link equation, Equation 3 [3].

$$P_R = P_T + G_T + G_R - Losses \quad \textbf{(3)}$$

where:
$P_T$ = Power transmitted
$G_R$ = Gain of the receiver
$G_T$ = Gain of the transmitter

The losses in Equation 3 are a function of the distance between the antennas, the environment through which the wave propagates and the characteristics of the hardware used to process the signal. Increasing the gain of an antenna at a specified frequency can be accomplished by making it larger, as the effective area of a reflector type antenna is proportional to the physical dimension, Equation 4 [3].

$$G_R = \frac{A_{eff} * 4\pi}{\lambda^2} \quad \textbf{(4)}$$

where:
$A_{eff}$ = Effective aperture
$\lambda$ = Signal wavelength

In summary, a larger antenna reflector allows for more energy to be collected, increasing the energy per bit. Increasing the energy per bit allows for more bits per symbol or a shortening of the bit period, while maintaining an effective BER. Thus, all things being equal, a larger reflector means higher gain.

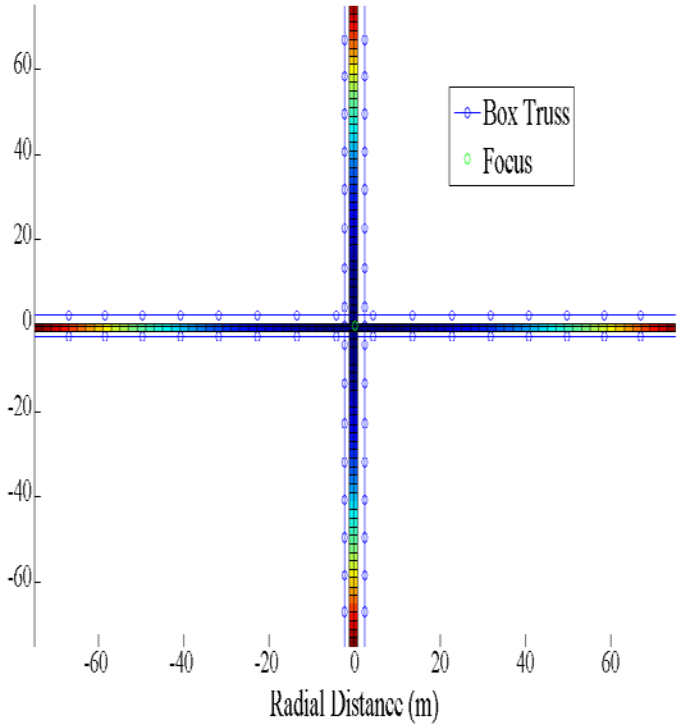
### *1.1.2 Sparse Aperture Antennas*

With very large antennas, packing them into a launch fairing becomes an issue. For comparison, the TDRS-L with its two 15-foot diameter, filled aperture, antennas launched inside an Atlas V 4-meter fairing [4]. The AstroMesh 12.25 m filled reflector, with nearly 4 times more collection area was also launched in a 4-meter fairing [5, 6]. The goal for the antenna studied here was to have similar collection area to a 50 m filled aperture parabolic reflector, calculated with Equation 5.

$$A_{collection} \approx \pi * r^2 = \pi * 25^2 = 1963 m^2 \quad \textbf{(5)}$$

where:
$A_{collection}$ = Antenna collection area for filled apertures
$r$ = Antenna radius

One can imagine the packing problems associated with going from AstroMesh's 113 square meters of collecting area to 1,963 square meters. A previous study developed a 150 m diameter, sparse-aperture, cross-shaped box truss design that will fit inside a 5-meter rocket fairing [7]. The design has a useful collection area of 1,964 square meters, as seen in Figure 1.




**[7]**

**Figure 1: Flat Reflector Box Truss**

### *1.1.3 RMS Surface Error*

As shown earlier, the amount of power an antenna receives greatly influences the effectiveness of the communication link. The wave nature of the electromagnetic (EM) energy requires that it arrive at the receiver in a constructive manner. This effect is easily seen in Figure 2, a simple example using two sine waves. The larger the phase difference between the two waves the less total energy that will arrive at the focal point.

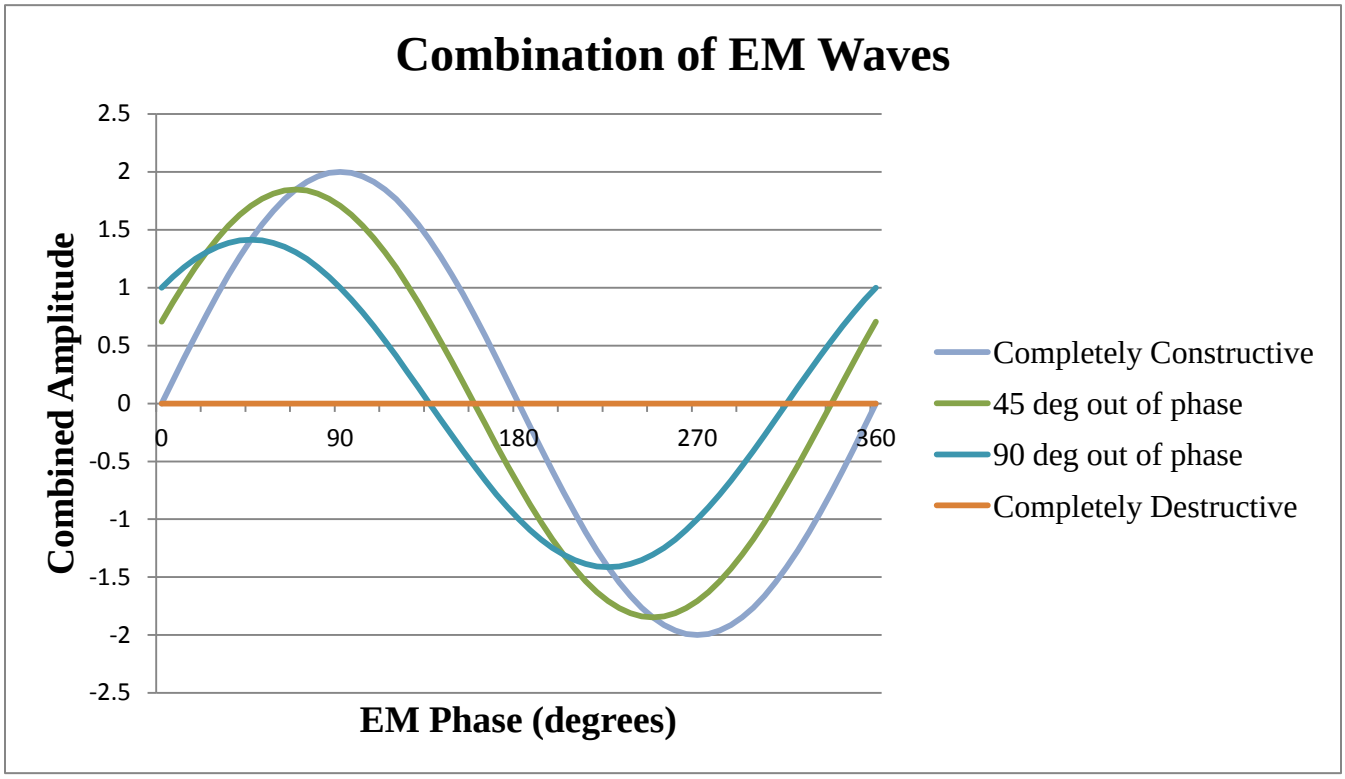



**Figure 2: Comparison of Interference in EM Wave**

This study is using a maximum error of 20% in the wavefront. This leads to a loss in maximum signal power from an ideal parabolic reflector of less than 20%, as seen in Figure 3. Due to the antenna being a reflector type, the 20% error is halved when

translated to a normal surface error. The antenna will operate in the L-band at a frequency of 1.5 - 2 GHz, or 15 - 20 cm wavelength. At this frequency the maximum Root Mean Square surface error ($\varepsilon_{RMS}$) is 15 mm.

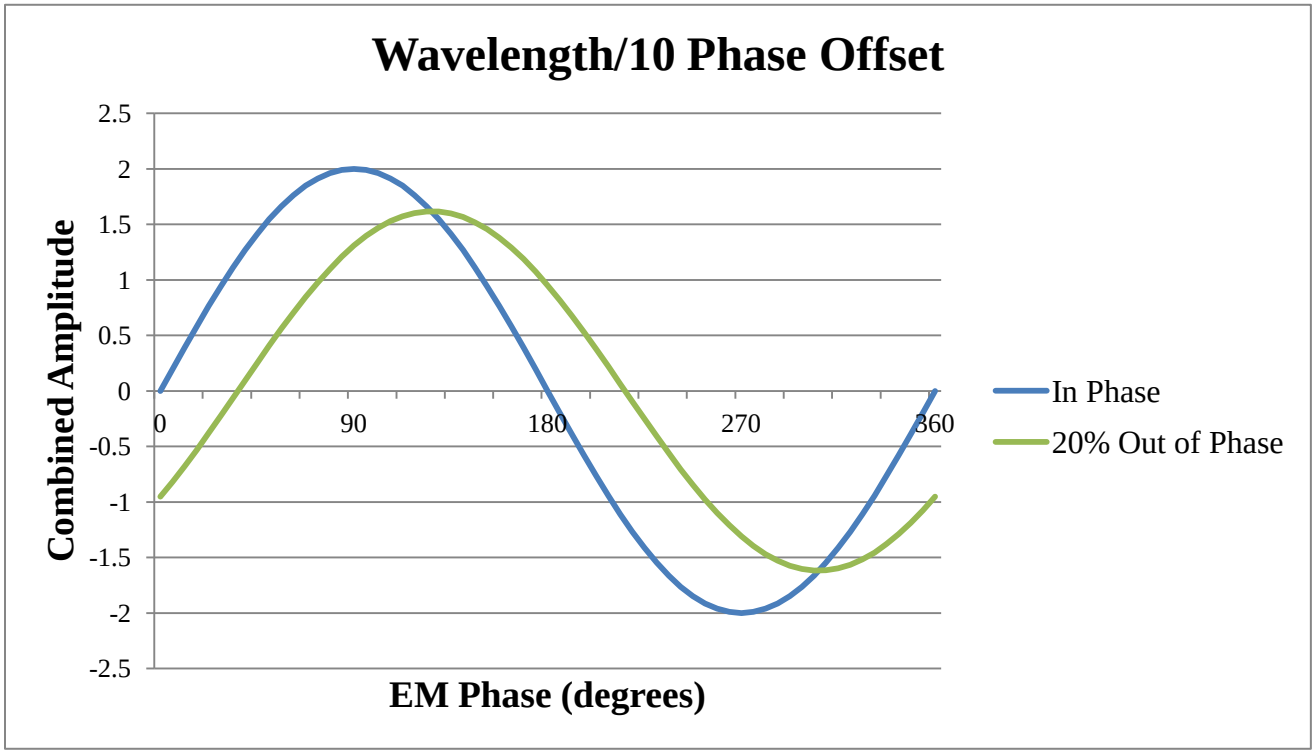



**Figure 3: Acceptable Phase Offset**

### 1.2 Research Objectives

The objective of this research is to continue exploring the technical basis for developing and deploying very large sparse aperture antennas. Specifically, answering the following questions:

- What environment will the satellite be operating in?
- What is the effect of fabrication errors on the proposed structure?

- What is the remaining error budget for the undefined portion of the system?

In short, are these systems technically feasible?

### 1.3 Materials & Equipment

The bulk of the work for this study was accomplished using software among the standard AFIT applications. MATLAB® (R2012a) was used for development of the ideal truss structure and reflector geometries. All anomalies to the initial geometry were constructed in the program, as well as development of Finite Element Analysis (FEA) cards. Static FEA was performed in NASTRAN (10.2), with resulting structures viewed in FEMAP 10.2. The final analysis of reflector errors was then performed back in MATLAB®.

### 1.4 Scope

The flat cross-shaped box-truss reflector from a previous study was modified to achieve the desired parabolic shape. This was accomplished through a geometric analysis of each truss section solved iteratively outward from the central hub. In the static FEA standard material properties for Kevlar carbon fiber are used, with tension members modeled as shortened bars. This ideal model was used for comparison when calculating the surface error of the reflector.

To produce more realistic models, errors in the structure were induced using a uniform distribution, with extreme values dictated according to desired manufacturing accuracy. Errors were induced in the structure through various members to ascertain their contribution to the overall error. After compiling hundreds of models, a final

analysis was completed against the ideal model. This produced expected extreme truss deviations and average surface errors throughout the structure.

### 1.5 Overview

This thesis is arranged with Chapter 2 as a literature review of published techniques related to and used in this research. The literature review is in two sections, section one being a look at past work completed on the geometry of the cross-shaped reflector. The second section, 2.2, is an examination of surface normal error in filled-aperture and flat strip antennas from various sources.

Chapter 3 contains information on the environment and operational considerations of the satellite; Chapter 4 presents the methods used in applying the literature review and creation of models; Chapter 5 holds the results and analysis of the geometric error budget; and Chapter 6 contains a review, conclusions and future work required.

# II. Background & Literature Review

This chapter will cover past research related to this specific effort and concepts involved in solving for the surface errors in reflector antennas. The preliminary geometry and mechanisms for the flat reflector truss structure were largely determined by recent efforts at AFIT [7]. Electrical properties of the idealized parabolic structure were addressed in an AFIT thesis from 2013 [8]. Estimation techniques for environmental effects in satellite drift and disturbance torques are explored to further understand sources of error in the system [1]. Research into the solution for surface errors in large antenna structures spans the last 30+ years. The work explores how errors propagate through the underlying truss and what accuracies are achievable with precise manufacturing and no active correction devices [8, 9]. The methods and models in these documents are the basis for this research.

## 2.1 Geometry of Cross-Shaped Flat Reflector

This work stems directly from the development of the initial design for a cross-shaped sparse-aperture reflector type antenna [7]. A sparse-aperture, not a full 50 m filled paraboloid, but simply a cross whose arms take the shape of a parabola. The member dimensions are given in Table 1.

**Table 1: Cross-Shaped Sparse Aperture Truss Members**

| Type | Length (m) | Outer Diameter (mm) | Thickness (mm) |
|---|---|---|---|
| Upper Longeron | 9.2126 | 51 | 0.635 |
| Horizontal Batten | 8.74 | 51 | 0.635 |
| Vertical Batten | 4.7 | 51 | 0.635 |
| Diagonal Wire (side average) | 10.3422 | 3 | 3 |

As shown in Figure 1, the reflector truss has four arms extending from a central hub with the focus at 80 m. Each arm is composed of eight truss bays each, seen in parabolic configuration in Figure 4. The longerons extend radially at the top and bottom of each bay. The vertical battens extend through the depth of the truss and diagonal wires connect the joints on each face. Only one exterior diagonal wire on the front and rear faces of each bay is shown for clarity in. The horizontal battens round out the remainder of each bay [7].

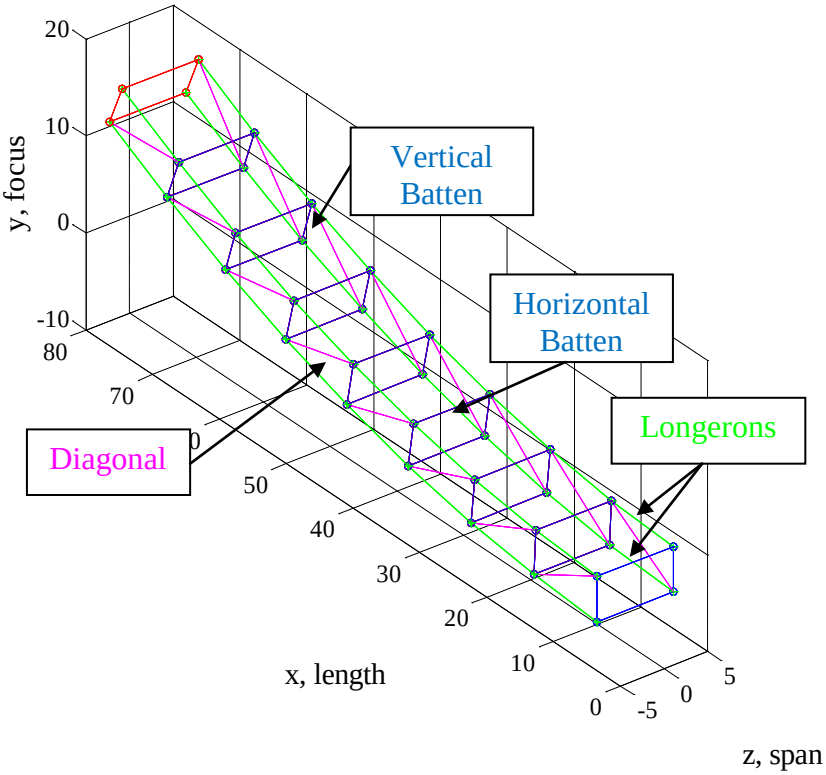



**Figure 4: Parabolic Reflector Box Truss**

Each longeron and horizontal batten is hinged in the middle for compact storage. The hinge locks in place once a straight configuration has been reached. Each node between truss members is a pin joint, allowing only motion about one axis. This will be important later in the analysis section.

The reflector surface itself is a knitted mesh material with a density of 50 $g/m^2$ [9]. Two shapes were considered for the panels and rectangular ones were chosen over triangular for their ability to maintain their shape under shear loads that may be introduced by a proposed stiffening system. Tessellation patterns were proposed based on a high and low end for the surface $\varepsilon_{RMS,}$ and panels not extending across truss bays. This results in two patterns using rectangular panels with the characteristics in Table 2. This paper will use the second tessellation pattern for analysis.

**Table 2: Mesh Tessellation Pattern**

| Rectangles per bay (axial x transverse) | Radial Length (m) | Horizontal Length (m) | # Radial | # Horizontal | $\varepsilon_{RMS}$ (mm) |
|---|---|---|---|---|---|
| 3x4 | 2.278 | 2.25 | 31 | 3 | <3 |
| 2x3 | 3.071 | 3.37 | 23 | 2 | <6 |

## 2.2 Surface Errors in a Large Space-Based Antenna

Reflector accuracy has three different error sources: errors in the reflector surface, errors due to imprecise beam pointing, and errors from a poorly positioned feed horn [10]. This research will focus on the required and achievable limits of structural reflector surface accuracy. Looking merely at passive support structures, this research uses a 10% maximum offset in the wavefront. This is in line with previous work that deemed a range

of 4-12% maximum error in the EM wavefront was acceptable for similar communication satellite applications [9].

The diameter-to-wavelength ratio of 1,000 is comparable to other satellite missions studied ranging between 1,000 and 100,000 [9]. The 10% error corresponds to a Root Mean Square (RMS) surface error of 15mm ($\lambda/10$), leading to an overall diameter-to-surface-error ratio of 10,000. This figure is comparable to recently deployed hoop-mesh antennas and conservative compared to those in the Large Space Systems Technology Program requiring ratios up to 200,000 [5, 9]. Noting the configuration being studied is within bounds of the theoretically possible, shown in Table 3, we can move deeper into the study confidently.

**Table 3: Diameter to RMS Surface Error**

| **Satellite** | **Frequency (GHz)** | **Diameter/ $\varepsilon_{RMS}$** |
|---|---|---|
| THURAYA/GEM | 1.6 | 9,423 |
| INMARSAT | 1.6 | 9,000 |
| MBSAT | 2.66 | 9,230 |
| **Sparse Reflector** | 2.0 | 10,000 |

The first stop for further study is the rectangular mesh chosen for the design. The mesh has very little stiffness and is thus assumed in uniform tension throughout. The error between a nearly-flat, tessellated mesh surface and a spherical reflector is given by

[9]. This formulation can be used to find the maximum allowable rectangular mesh size from a given $\varepsilon_{RMS}$ allocated from the overall error budget in a filled aperture.

$$\frac{a}{D} = 7.33\sqrt{\frac{F}{D}\left(\frac{\varepsilon_{RMS}}{D}\right)_{allow}}\left[1+\left(\frac{b}{a}\right)^4\right]^{-1/4} \quad \textbf{(6)}$$

where:
$a$ = Larger rectangular dimension of mesh facet
$D$ = Reflector diameter
$F$ = Focal length (2 x radius of curvature of the sphere)
$b$ = Shorter rectangular dimension of mesh facet

A second major source of error in a reflector is the accuracy of the underlying truss. Most of the analyses used structures with pin joints, allowing them to treat all errors in a truss as errors in the length of individual members [11]. For a spherical pre-tensioned truss, similar to the cross-shaped parabolic reflector, the surface error becomes a function of the error in the diameter and errors in the rim, Equation 7 [9].

Equation 7 is derived from a comparison of error analysis to natural vibration analysis [12]. Thus the error can be allocated throughout the truss structure, and the required manufacturing error determined. Standard deviations in manufacturing accuracy are on the order of $10^{-3}$, but if necessary $10^{-6}$ is achievable [9]. This reference was extended to an examination of how antenna surface error is related to not only truss member errors, but the number of truss bays in the structure, seen in Equation 8 and Equation 9 [11].

$$\frac{\varepsilon_{RMS}}{D} = \sigma_{\varepsilon}\sqrt{\frac{D}{l_b}\left(0.060\frac{l_b}{H}\right)^2 + \left(\frac{\varepsilon_{diam}}{D}\right)^2 + \left(\frac{\varepsilon_{rim}}{D}\right)^2} \quad \textbf{(7)}$$

where:
$\varepsilon_{RMS}$ = RMS surface error from truss errors
$\sigma_{\varepsilon}$ = Standard deviation of the unit length error for truss elements
$l_b$ = Radial length of a truss bay
$D$ = Reflector diameter
$H$ = Height of truss bay
$\varepsilon_{diam}$ = RMS error in truss diameter
$\varepsilon_{rim}$ = RMS error in truss rim

$$\frac{\varepsilon_{diam}}{D} = \frac{1}{2}\sqrt{\frac{l_b}{D}}\left(\frac{H}{l_b} + \frac{1}{4} * \frac{l_b}{H}\right) \quad \textbf{(8)}$$

Variables as above

$$\frac{\varepsilon_{rim}}{D} = 0.177\left(\frac{D}{L} + \frac{L}{D}\right) \quad \textbf{(9)}$$

where:
$L$ = Total length of truss

There is a marked difference between this filled aperture pretensioned-truss configuration from Reference [9] and the sparse aperture examined in this research. The only non-tension members in the filled aperture configuration are a central column and a compression ring, all other components are tension members. Thus a new approach is needed to define how errors build up in the sparse aperture cross from errors in its

members. Not only is there no outer ring, which dominates the error terms in the filled aperture, but the tubular radial members have the greatest contribution from the underlying truss to surface error.

The surface error in a flat triangular truss structure was investigated by Greschik et al and correlated to the total number of bays in a truss [11]. This relation is of obvious concern to an investigation of very large antennas as they necessarily have more bays to fit in a given launch fairing. The result is not an exact analog to the cross-shaped parabolic box truss design, but their methodologies are applicable. A uniform distribution of random errors was induced in the length of each member of the truss structure. The number of truss bays for the structure was then varied, over a constant length of the structure. With configurations ranging between 1 and 100 bays, 35,000 to 3.5 million simulations were run to use in the surface error calculations. With the length data, the connection points for the reflector surface are known. Each bay in the truss is covered by a bi-linear panel, which is distorted from a perfectly flat shape based on the errors in the underlying truss structure. The RMS surface error was treated as half the offset from ideal, calculated with Equation 10.

$$\varepsilon_{RMS} = \sqrt{\frac{\int \left( u_{srf} - u_0 \right)^2 dA}{4A}} \tag{10}$$

where:
$\varepsilon_{RMS}$ = Average RMS surface error
$u_{srf}$ = Surface normal to actual antenna position
$u_0$ = Surface normal to ideal antenna position
$A$ =Surface area

Reference [11] concludes with a relationship between the maximum error in a truss member and the total number of bays in a truss structure. The analysis section will show the results of this thesis will show this as a good initial estimate. However, the method of integrating the surface errors does not apply to a parabolic reflector. The integration would provide an unhelpful error volume, contrary to the required surface normal error which causes a wavefront error. A new method of calculating the surface normal error for any shape of reflector will be developed in section 4.4 of this text.

**2.3 Summary**

The research in this section has shown the overriding concern for an antenna, from a mechanical stand-point, is the accuracy of the reflector surface. A very large antenna will have more contributions from the underlying truss to surface errors. Potential sources of error and methods for solving the surface geometry of the reflector mesh have been explored for filled aperture reflectors, and contrasted with the needs for a sparse aperture reflector. There have also been numerous examples of first order evaluations that suggest the structure under consideration is technically possible.

## III. Operational Considerations

Previous research by Wilson has shown this design has similar electromagnetic performance characteristics to a 50 m filled aperture antenna, with a gain of greater than 50 dB and 3 dB beamwidth of 0.8 degrees [8]. Operating at geostationary orbit (GEO) 35,786 km above the equator, the antenna radiation pattern on Earth's surface will have a diameter of 500 km, Equation 11.

$$D = 2 * \tan\left(\frac{BW}{2}\right) * alt \tag{11}$$

where:
$D$ = Beam diameter at Earth's surface
$BW$ = 3dB beamwidth of radiation from antenna
$alt$ = Altitude of spacecraft above Earth's surface*

Due to the tight pointing accuracy required by the 0.8 degree 3 dB beamwidth, it is important to consider the forces that will affect the spacecraft once it is on orbit. This will drive the design of an attitude determination and control system, which will naturally alter the geometry of the antenna during pointing maneuvers. First, this research will explore the amount of delta V the spacecraft will have to expend throughout its lifecycle to maintain its desired location. Secondly, especially with such a large structure, it is necessary to see how uneven forces will cause disturbance torques.

* Curvature of the Earth is ignored due to negligible effects at this length

### 3.1 Perturbation Drifts

Orbit perturbations at this altitude are mostly due to gravitational interactions with the Sun and Moon, which would cause the inclination of the satellite to change [1]. These north-south drifts are given by Equation 12 and Equation 13. An inclination of roughly 0°, and angle relative to the Sun and Moon of $\alpha \approx \gamma \approx 23°$ will result in a worst case annual $\Delta V$ requirement of 51.38 m/s [1].

$$\Delta V_{Moon} = 102.67 * \cos(\alpha)\sin(\alpha) \quad \textbf{(12)}$$

where:
$\alpha$ = Angle between orbit plane and the Moon's orbit

$$\Delta V_{Sun} = 40.17 * \cos(\gamma)\sin(\gamma) \quad \textbf{(13)}$$

where:
$\gamma$ = Angle between orbit plane and the Sun's orbit

The next largest effect at this altitude is caused by the Earth not being a perfect sphere. This translates into an east-west drift which is a function of how far away from a stable longitude the satellite is orbiting [1]. Using the closest stable location as 105° West, and a desired orbit over Dayton, OH the $\Delta V$ for this drift is ~1.14 m/s per year, calculated by Equation 14.

$$\Delta V_{J22} = 1.715 * \sin\left(2 * \left|\Lambda_D - \Lambda_S\right|\right) \quad \textbf{(14)}$$

where:
$\Lambda_D$ = Desired longitude
$\Lambda_S$ = Nearest stable longitude

In total, Table 4 contains the most important drifts the satellite is expected to experience at GEO. These are similar to any GEO satellite, as they are a function of the satellites orbit, not configuration.

**Table 4: Station Keeping ΔV Requirements**

| **Body** | **Direction** | **Annual ΔV (m/s)** |
|---|---|---|
| Moon | N-S | 36.93 |
| Sun | N-S | 14.45 |
| J22 | E-W | 1.14 |

### 3.2 Disturbance Torques

To begin with, let us examine the torques from solar radiation pressure. Due to the structure being a truss, and the mesh a loose wire network, the incident surface area is quite small, and there are not many scenarios where this should pose a problem. There are two extreme scenarios for this nadir pointing craft: 1) emerging from eclipse with one arm and half of the perpendicular arms illuminated on one side, 2) nadir pointing with the feed-horn structure illuminated on one side. The torque can be calculated by the incident solar radiation perpendicular to a surface area, seen in Equation 15 [1].

$$T_S = \frac{\Phi}{c} A_s \left(1+q\right)\left(cp_s - cm\right)\cos\left(\varphi\right) \quad \textbf{(15)}$$

where:
$T_S$ = Torque on the center of mass
$\Phi$ = Solar Constant (1366 W/m$^2$)
$c$ = Speed of light (3E8 m/s)
$A_s$ = Surface area
$q$ = Reflectance, taken as 0 for carbon fiber structure
$cp_s$ = Solar center of pressure [0,0,19.19]
$cm$ = Center of Mass [0,0,0]
$\varphi$ = Angle of incidence of the sun (90° worst case)

These forces are nearly an order of magnitude larger than they would be for a filled 50 m reflector using compression ring type construction, see Table 5. That structure is mostly tension cables with minimal cross section, and the reflective compression ring is at most 25 m from the center of mass, as opposed to outer members for the sparse aperture at over 70 m. Despite the larger forces for the very large sparse aperture reflector, the worst case solar torque amounts to a few milli-Newtons which is easily overcome by a properly sized attitude determination and control system (ADCS).

Next are forces due to the gravity gradient, which occur when the center of gravity does not align with the center of mass [1]. The Moments of Inertia (MoI) are estimated by considering only the antenna structure and mesh surface as the major contributing mass elements. The parallel-axis theorem is used for the thin-walled cylinders of the truss members, and the mesh is taken as a uniform density flat plate.

The resulting gravity gradient torque is 2 orders of magnitude larger for the sparse aperture reflector than then 50 m filled aperture reflector, see Table 5. The torques are

quite small and result in angular accelerations that oscillate the structure around a point where the antenna feed horn is pointing nadir, Equation 16.

$$T_g = \frac{3\mu}{2R^3}\left|I_y - I_z\right|\sin\left(2\theta\right) \quad \textbf{(16)}$$

where:
$T_g$ = Torque on the center of mass
$\mu$ = Earth's gravitational constant (3.99E+14 $m^3/s^2$)
$R$ = Orbit radius to center of Earth (4.22E+7 m)
$I_z$ = Moment of inertia about the z-axis (6.11E+06 kg*$m^2$)
$I_y$ = Moment of inertia about the y-axis (5.23E+06 kg*$m^2$)
$\theta$ = Nadir off-axis angle (45° worst case)

Lastly, to address the maximum magnetic torque, a relation between the orbit, the Earth's magnetic field and the spacecraft's magnetic state is required, Equation 17 [1]. Using NASA design guidelines for estimating magnetic dipole for the sparse aperture antenna based on weight results in a very low value for the magnetic torque. The sparse aperture values are only 10% larger than the values for a 50 m filled aperture reflector, see Table 5.

$$T_m = \Psi\left(\frac{M}{R^3}\tau\right) \quad \textbf{(17)}$$

where:
$T_m$ = Torque on the center of mass
$\Psi$ = Spacecraft's dipole, 7.62 A*$m^2$ [13]
$M$ = Magnetic moment of the Earth
$R$ = Orbit radius to center of Earth
$\tau$ = Scaling factor for latitude of satellite, 1.2 at equator

Solved for the worst case scenarios, a properly sized ADCS should easily be able to compensate for the environmental torques discussed in this section, and consolidated in Table 5.

**Table 5: Max Environmental Torques**

| Axis | MOI ($kg*m^2$) | Gravity Gradient (Nm) | Solar Radiation Pressure (Nm) | Magnetic Moment (Nm) |
|---|---|---|---|---|
| X | 6.11E+06 | 7.06E-03 | 7.14E0-3 | 9.52E-07 |
| Z | 6.11E+06 | 7.06E-03 | 7.14E0-3 | 9.52E-07 |
| Y (nadir) | 5.23E+06 | NA | NA | 9.52E-07 |
| 50 m filled Y | 3.46E+04 | 4.00E-05 | 9.36E-04 | 8.66E-08 |

### 3.3 Summary

The initial station keeping budget is 51 m/s per year, in line with other satellites orbiting at GEO. The environmental torques are larger in all cases for the sparse aperture antenna than the 50 m filled aperture antenna. This is due to a larger structure having greater mass away from its center of mass. Even though the forces are larger, they are on the order of milli-Newtons. The first order analysis values are small enough to not be a major concern at this point. Future work to determine the operational suitability of the antenna system will require further analysis in countering the environmental effects.

## IV. Methodology of Error Determination

This section will consider an analysis of the geometry used to create the ideal truss structure, which is used to gauge errors in the location of its connection points to the parabolic reflector mesh. All trusses with errors in their members are compared to the ideal truss. Next the method used to induce uniform errors into the truss members will be explored, and how the models were set-up for FEA. With the results of many different perturbed models, a solution for the normal surface error is addressed using Lagrange multipliers. All of this information will be used for the computation of the $\varepsilon_{RMS}$ of the reflector mesh.

### 4.1 Analysis of Geometry

The geometry of the flat cross-shaped structure in Figure 1 is a constraint on the extension to a parabolic shape. The members from Table 1 are arranged in four arms, each containing eight bays. The focus is at 80 m from the apex of the paraboloid. Using this information an ideal truss was constructed beginning with the simple parabolic equation, Equation 18.

$$y = \frac{x^2 + z^2}{4p} \tag{18}$$

where:
$p$ = Distance to focus from parabola apex (80 m)

This produces the ideal shape for a 150 m diameter parabolic reflector seen in Figure 1. Notice the rise in the parabola from the central truss to the end of one arm of over 17 m. The underlying truss will have to be configured in such a way that it follows this curvature.

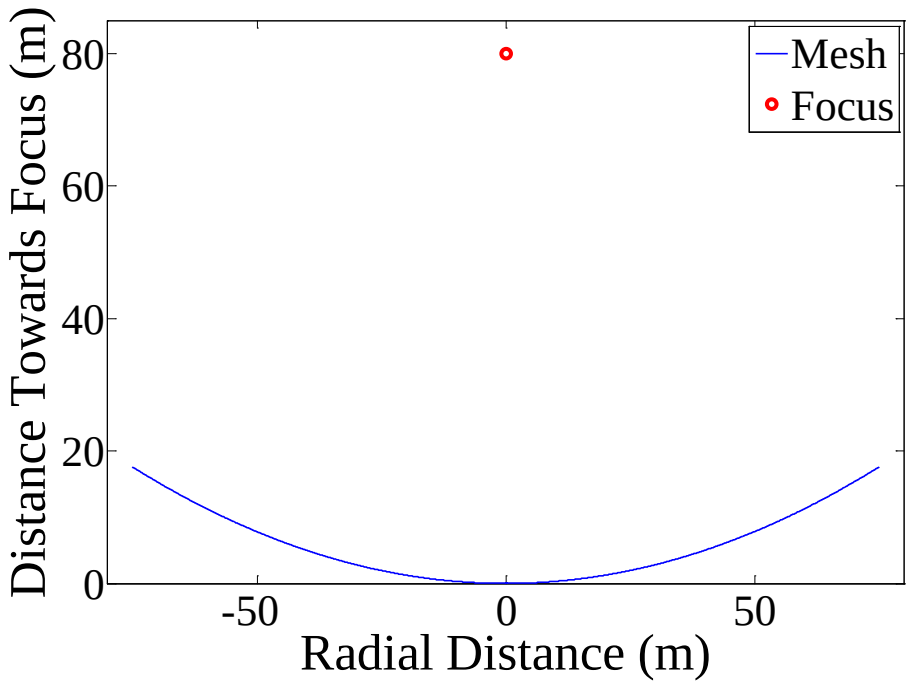



**Figure 5: Ideal 2D Parabolic Reflector**

The curvature in the mesh surface in one arm, as seen looking toward the center of the structure from the end of one arm is shown in Figure 6. Throughout the mesh this same curvature of roughly 4 cm occurs. The shape is important in Chapter 4 when examining the positioning of the mesh panels that make up the actual reflector surface, instead of the idealized paraboloid.

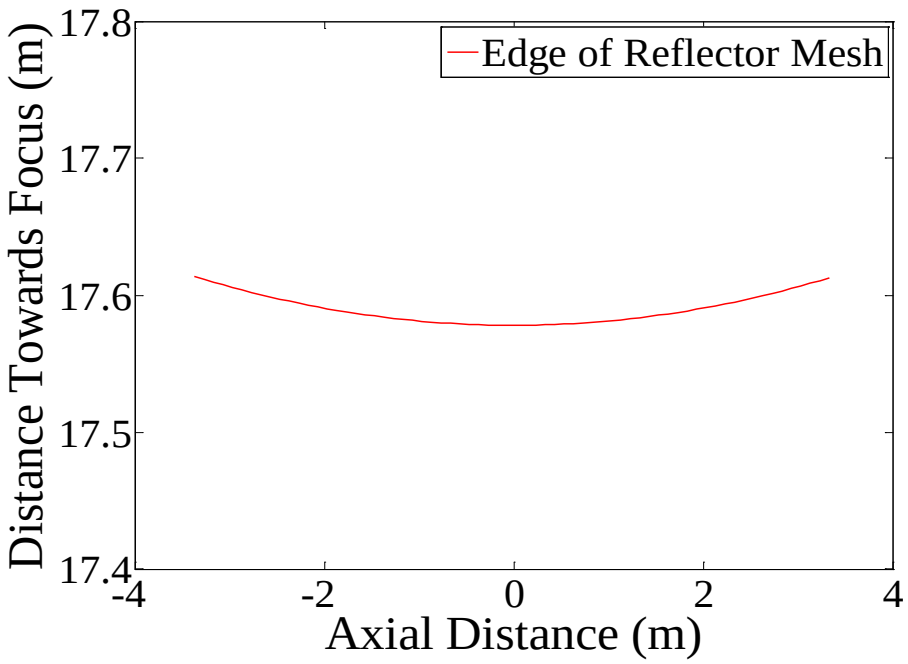



**Figure 6: Axial Curvature in Mesh**

To correlate the underlying truss to the shape of the parabolic mesh, and make an ideal truss, the longerons closest to the mesh and all battens were held as constants. The following formulation, Equation 19, solves for the nodes in the structure using the parabolic equation and longeron constraint in the Pythagorean Theorem.

$$l_L{}^2 = \left(x_n - x_{n-1}\right)^2 + \left(\frac{x_n^2}{4p} - \frac{x_{n-1}^2}{4p}\right)^2 \quad \textbf{(19)}$$

where:
$l_L$ = Length of an upper longeron (9.2126 m) [7]
$x_n$ = X-position of next node in the structure radially
$p$ = Distance to focus from parabola apex.(80 m)

Solved iteratively for $x_n$, this provides the (x,y) coordinates for the nodes connecting each upper longeron. Horizontal battens simply offer an offset from the center of the truss for transition into the three dimensional model. Thus the entire top geometry of the truss is solved. To extend the structure to the bottom geometry, the angle from local horizontal for each batten was used, and vertical battens are initially assumed to form right angles with the upper longerons, Equation 20. This will not produce the 'ideal' truss desired as the diagonal tension members will change the shape of each bay, but is close enough for an initial analysis.

$$x_{bot\_n} = \left( x_{top\_n} - x_{top\_n-1} \right) + b_v * \sin\left( \cos\left( \frac{x_{top\_n} - x_{top\_n-1}}{l_L} \right)^{-1} \right) \quad \textbf{(20)}$$

where:
$x_{bot\_n}$ = X-position of next lower longeron node radially
$x_{top\_n}$ = X-position of next upper node in the structure radially
$b_v$ = Length of vertical batten (4.7 m)
$l_L$ = Length of an upper longeron (9.2126 m)

With Equation 19 and Equation 20 all of the nodes of the structure can be solved for in the ideal truss. With the lower longerons longer than the top, a curve to the structure is achieved without the added complexity of locking joints or the like. All future perturbations will be checked against this design. The result is the ideal slender truss structure with mesh surface seen in Figure 7.

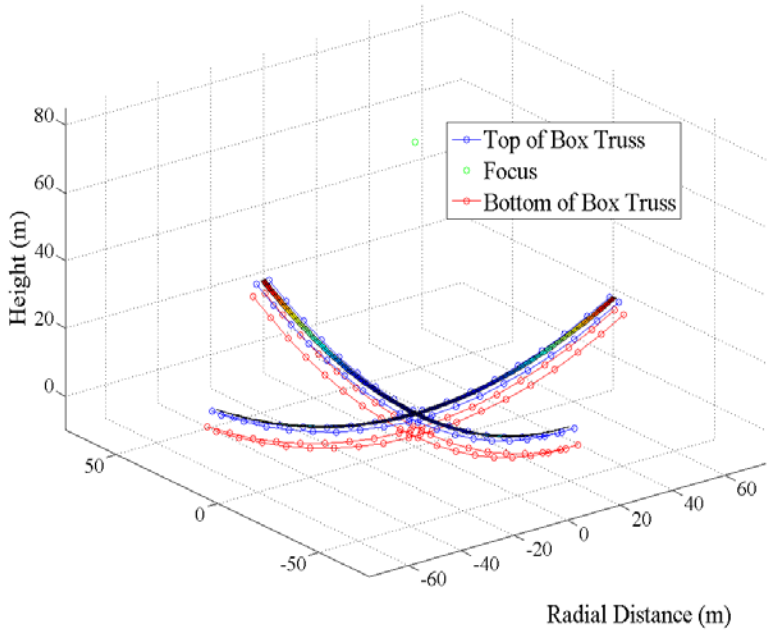



**Figure 7: 3D Reflector Box Truss**

### 4.2 Inducing Errors in Truss Members

With the ideal structure solved for, it is necessary to perturb the design to understand the effects on the mesh surface of errors in the underlying truss. Following similar past examples, a uniform distribution of errors bounded by the achievable manufacturing accuracy will be introduced [12, 11]. Solving the geometries for the perturbed truss requires a different methodology, no longer able to rely on the upper-longer and subsequent vertical batten forming a right angle.

The primary reasoning for the approach used was to keep it in line with how errors would develop in a manufactured antenna. Similar to the iterative approach used in the ideal model, each truss arm is calculated from the central truss outward by bay. The inputs to the model are the locations of each corner of the central truss, the lengths of

each truss member and two opposite diagonal cable lengths. The errors in each truss take the form in Equation 21, using MATLAB's® random number generator, that pulls from a uniform distribution.

$$truss\_error = -\max\_error + 2 * \max\_error * rand() \quad \textbf{(21)}$$

where:
*truss_error* = Proportional error in the length of a truss member
max_*error* = Proportional error allowable by chosen manufacturing accuracy
*rand()* = Random scalar chosen from the uniform distribution (0,1)

Estimates for diagonal member lengths are solved first at the end of the arm closest to the center of the antenna. Any deformation in the truss bay is propagated to the next radial bay. Using the configuration in Figure 8, Equation 22 solves for a diagonal cable length using the law of cosines and the current and past longerons' deflections from horizontal assuming deflections remain planar. Remaining interior angles are solved for the given lengths until a solution converges.

$$l_{diag1,n} = \sqrt{l_{bat1}^2 + l_{bat2}^2 - 2l_{bat1} * l_{bat2} * \cos\left(\alpha_n + \frac{\pi}{2} - \alpha_{n-1}\right)} \quad \textbf{(22)}$$

where:
$l_{diag1}$ = Diagonal cable length
$l_{bat1}$ = Length of inner vertical batten
$l_{bat2}$ = Length of outer vertical batten
$\alpha$ = Angle of upper longeron from horizontal

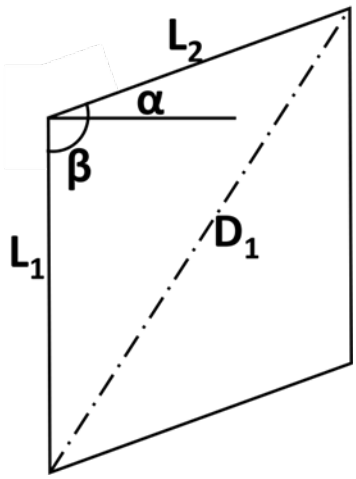



**Figure 8: Box Truss Bay Face**

To extend this solution and allow three-dimensional deflections requires the central truss node points, member lengths with errors and estimated diagonal cables be run through a similar solver. Initially two outer, opposite, corners are solved for, using the constraint that outer nodes can only be separated from inner nodes using the length of their connecting members. Subsequent truss nodes are calculated using the MATLAB® non-linear least square solver 'lsqnonlin.' Once automated, the code allows for the computation of a number of models with random perturbations scattered throughout the members.

### 4.3 Static FEA

Once the location of all the truss nodes are known, the pre-tensioned truss models are subjected to a FEA to find the final static shape. Any difference in length of the truss members from the no-error size will shift the node locations. Shifted node locations translate into a shift in the edge of the reflector mesh, where it is connected to the truss.

The new location of the mesh is then compared to its normal offset from the ideal parabolic mesh, which gives the surface error.

The NASTRAN solver was used, due to the large size of an analytical solution. This required a definition of the material properties, in this case a standard carbon fiber Kevlar was used, per Table 6 [14]. With these properties and geometries, the stiffness's and MoI can be calculated in Table 7.

**Table 6: Carbon Fiber Properties**

| | | | |
|---|---|---|---|
| **Young's Modulus** | 30 GPa | **Density** | 1,400 kg/m$^3$ |
| **In-plane Shear Modulus** | 5 GPa | **Truss Diameter** [7] | 51 mm |
| **Poisson's Ratio** | .33 | **Truss Wall Thickness** | 0.635 mm |
| **Coefficient of Thermal Expansion** | 7.4 mm/(m*K) | **Cable Diameter** | 3 mm |

**Table 7: Truss Member Properties**

| | **Longerons** | **Horizontal Battens** | **Vertical Battens** | **Diagonal Cables** |
|---|---|---|---|---|
| **Polar MoI (m$^4$)** | 6.3726E-08 | 6.3726E-08 | 6.3726E-08 | 7.9522E-12 |
| **Torsional Stiffness (N*m)** | 1.3834E+02 | 1.4583E+02 | 2.7118E+02 | 1.5378E-02 |

The model also required a treatment of the cables such that they would be under a nominal 100 N of tension. This is achieved by greatly increasing the coefficient of thermal expansion (CTE) for the cable with respect to the other truss members, ~50 times greater. A $\Delta T$ is then chosen to produce the desired force in the cable, Equation 23.

$$\frac{F}{k} = \Delta T * CTE * l_{cable} \quad \textbf{(23)}$$

where:
$F$ = Desired force in cable
$k$ = Axial stiffness (AE/L)
$\Delta T$ = Change in temperature from no contraction in cable
$CTE$ = CTE in the cable, (352 mm/[m*K])
$l_{cable}$ = Length of the cable

Recovering the mesh connection points from NASTRAN brings us one step closer to comparison with the ideal parabolic reflector. Remember that the true error is calculated by the error in the normal direction. For instance, if a portion of the reflector is out of position axially, but at a constant radius, it will not produce an error in the signal path length. Thus a solution for the surface's normal deflection must be found.

### 4.4 Solution for Error Normal

The shortest vector between a point and a surface is also normal to the surface. Thus to find the surface normal for the error requires a minimization of the distance between the error point and the ideal surface. The method of Lagrange multipliers was used to arrive at this result, with the constraint that the point you are minimizing the distance to must lie on the surface, Equation 24 [15].

$$
\begin{aligned}
Lag\left(x,y,z,\zeta\right) &= d\left(x,y,z\right)+\zeta * g\left(x,y,z\right) \\
d\left(x,y,z\right) &= \left(x-x'\right)^2+\left(y-y'\right)^2+\left(z-z'\right)^2 \\
g\left(x,y,z\right) &= x^2+z^2-4p*y
\end{aligned}
\tag{24}
$$

where:
*Lag()* = Lagrangian formulation
*d()* = Relation attempting to minimize, length between true point and ideal point
$\zeta$ = Lagrange multiplier
*g()* = Constraint equation, ideal point must lie on the paraboloid surface
*(x,y,z)* = Ideal points of reflector
*(x',y',z')* = True points of reflector
*p* = Distance to focus from parabola apex (80 m)

To perform the minimization of the distance requires taking the partial derivatives of the Lagrangian. Setting them equal to zero is a necessary condition for a minimum, similar to finding the zero slope of a curve. This results in four equations and four unknowns that can be solved analytically in Equation 25.

$$
\begin{vmatrix}
\dfrac{\partial L}{\partial x}=2x\left(1+\zeta\right)-2x' & \dfrac{\partial L}{\partial y}=2\left(y-y'\right)-320\zeta \\
\dfrac{\partial L}{\partial z}=2z\left(1+\zeta\right)-2z' & \dfrac{\partial L}{\partial \zeta}=x^2+z^2-320y
\end{vmatrix}=0
\tag{25}
$$

where:
As in Equation 24

The final relation for normal error is as follows in Equation 26. Notice the singularities possible when solving for errors at the base of the paraboloid. This can be

avoided if there is only a zero value in either the x or z directions. For instance solve the quadratic for z instead of x if x=0, as their forms are similar. If both x and z are zero, the normal offset is obviously just equal to the y-value for the actual point of the mesh.

$$1+\zeta=\frac{x'}{x}$$
$$z=\frac{z'(x)}{x'}$$
$$y=y'+160\left(\frac{x'}{x}-1\right)$$
$$x^2\left(1+\frac{z'^2}{x'^2}\right)-320\left(y'+160\left(\frac{x'}{x}-1\right)\right)=0 \quad \textbf{(26)}$$

where:
As in Equation 24

This solution is illustrated in Figure 9. The red point is the physical location of a point on a distorted mesh surface. The blue surface is the desired ideal paraboloid mesh. The black line is normal to the ideal mesh surface and runs through the actual red point. The distance between the two is half the wavefront error, as this is a reflector antenna.

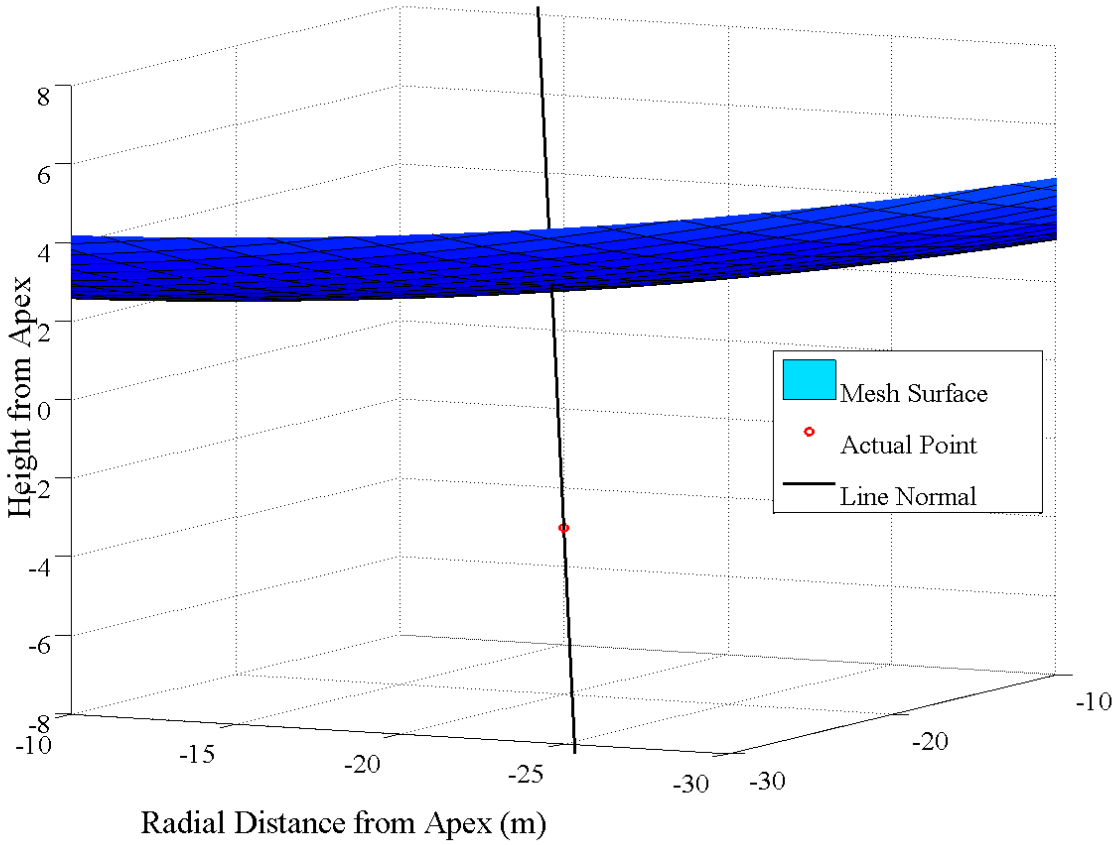



**Figure 9: Line Normal to Surface Through Actual Point**

When determining how truss errors result in mesh errors the $\varepsilon_{RMS}$ is calculated at offsets from each of the node points in the top of the structure. This produces the errors in the edge of the mesh at 64 points in each model. To determine how the tessellated reflector mesh pattern causes surface errors a single panel is compared to the shape of a parabolic mesh. A panel measuring 3.071 x 3.37 m has a $\varepsilon_{RMS}$ calculated every ten centimeters, for over 1,000 comparison points.

The $\varepsilon_{RMS}$ does not distinguish systematic errors, or biases in measurement. With relation to optical systems, each deviation can be thought of as a Zernike Polynomial

with a Noll index of 1 [16]. This piston error assumes all energy is radiated back towards the focus without any additional change in its path direction. The result is all of the reflected energy being captured at the focus, with only a wavefront error. The Zernike Polynomial for the reflector cannot be derived from the $\varepsilon_{RMS}$ value of the system. The algorithms used here don't assume null systematic error, but the $\varepsilon_{RMS}$ will give a bound for the allowable errors that will still produce a usable system. If, for instance, the focal length is not exact, the $\varepsilon_{RMS}$ budget can detail how out of alignment it can be.

### 4.5 Summary

This section started with a solution to the ideal parabolic truss. This truss was then perturbed with a uniform distribution of manufacturing errors and required a more robust solution allowing distortion in all axes. Next the material properties were explored prior to a static Finite Element Analysis. Lastly, the solution for normal error values from the FEA models was presented by the solution to a Lagrangian multiplier problem. This allows adjustments to the truss structure to be compared to the ideal reflector and its electrical properties.

# V. Analysis and Results

This chapter will begin with an analysis of the ideal truss model to a perfect paraboloid to measure its suitability for comparison. Next, perturbed trusses that have had errors induced in each of their respective members will have their surface normal errors determined with respect to the ideal paraboloid. With those errors understood, the error due to mesh tessellation is examined. These results will be compared to several references from Chapter 2 to make sure our outcomes are in line with expectations. Next the trusses will be perturbed by specific member type, for example only the top longerons. This will illuminate how each truss member contributes to the overall errors in the surface normal.

## 5.1 Analysis of Errors in the Truss Structure

The first step in the analysis involves comparing the ideal truss structure to the ideal paraboloid mesh. This will illuminate any errors in the underlying model. Due to there not being variations in the truss, one model was created and analyzed against the ideal mesh surface. The errors in the model are too small to see with the naked eye, Figure 10, but looking at the normal surface error distribution in Figure 11 brings a problem to light. There is clearly an offset in the model that causes an average surface error of 4.0958 mm. Until the model can be fully refined, this will have to be taken into account when considering errors in the perturbed trusses due to shifting of box geometry by tensioning cables.

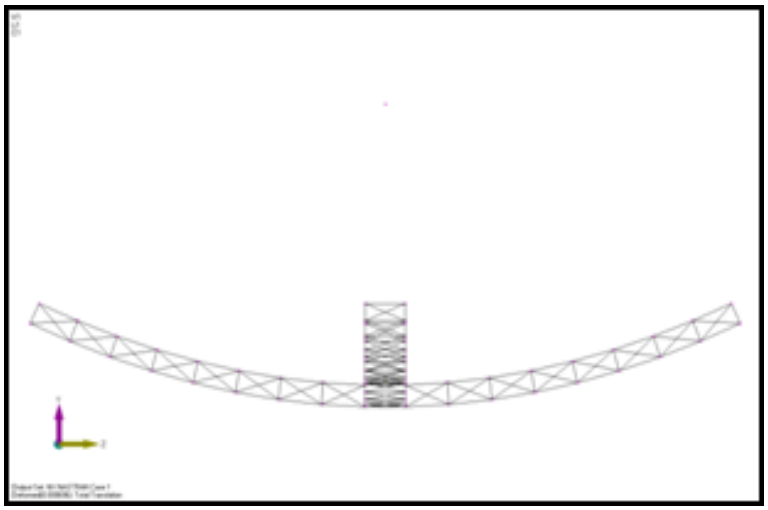


**Figure 10: Ideal Truss Structure**

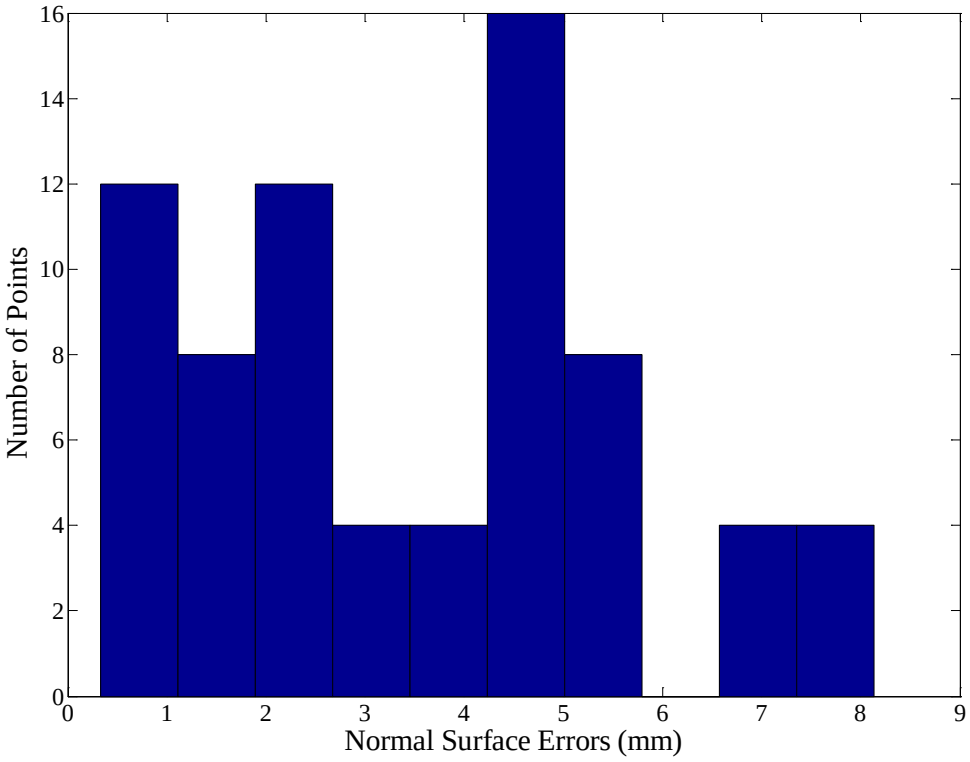



**Figure 11: Surface Normal Errors in Ideal Truss**

With knowledge of the constant error in the ideal model, analysis can extend to models with random errors induced in the truss members. As stated previously, errors were introduced in each individual type of member. Top longerons, bottom longerons, horizontal and vertical battens and diagonal cables, as well as a model with random errors in all members. Each of the 6 different configurations was run 30 times with a random error of $10^{-5}$ in proportional length. Run thru the NASTRAN FEA solver, each reflector has 64 error vectors, showing the difference between the ideal shape, and the varied shape. These 64 locations represent the locations where the reflector mesh is attached to the truss frame.

The first models have errors in all of the truss elements. The parameters for this analysis are in Table 8, and results in the right line of Figure 12. This allows a comparison to research in Chapter 2, ensuring the values are reasonable.

**Table 8: Analysis Parameters for Errors in All Truss Members**

| **Number of Models Analyzed** | **Absolute Error Allowed (proportional to member length)** | **Error Points per model (#)** | **Max $\varepsilon_{RMS}$ Observed (mm)** |
|---|---|---|---|
| 30 | 1E-5 | 64 | 4.7859 |

The surface errors are between 3.80 and 4.79 mm, while the mean is within 4% of the error in the ideal truss model. Removing the errors due to the ideal truss being non-ideal is a two step process justified because the nominal errors in the model can be removed through design and do not reflect manufacturing errors. The first step is to find the error vectors between the mesh connection points in the ideal model and a perfect paraboloid. These errors build up in the ideal truss even with no manufacturing error.

The second step is re-processing the surface normal data by removing the ideal truss errors from each node in the perturbed models. These corrected points then use the method of Lagrange multiples outlined in Equation 26 to find the surface normal error, and thus wavefront error. The result of reprocessing the data for models with errors in all their members yields the red line in Figure 12. The corrected values are between 0.6501 and 1.5228 mm, with a mean of 1.0619 mm. The shape of the line is more apparent, zoomed in for Figure 13.

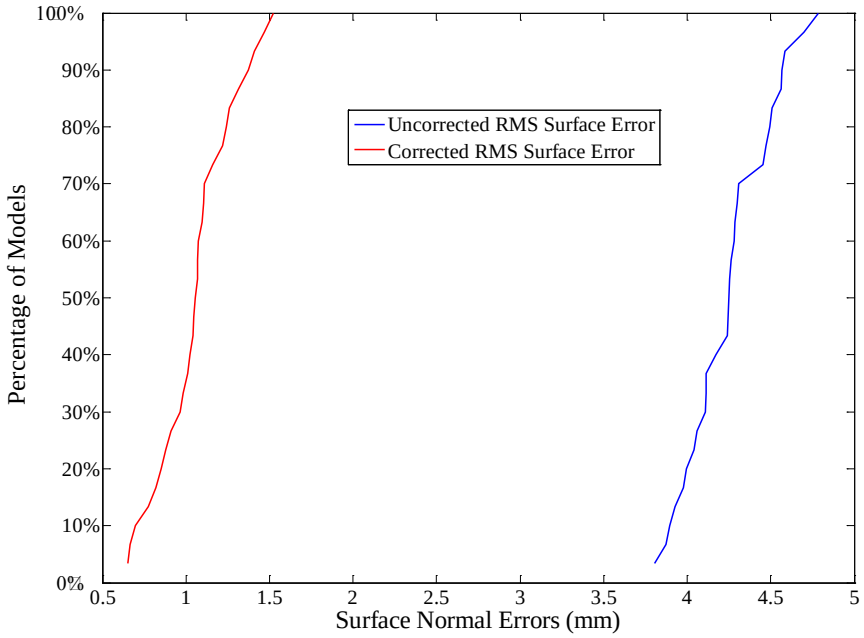



**Figure 12: Corrected & Uncorrected RMS Errors with Errors in All Members**

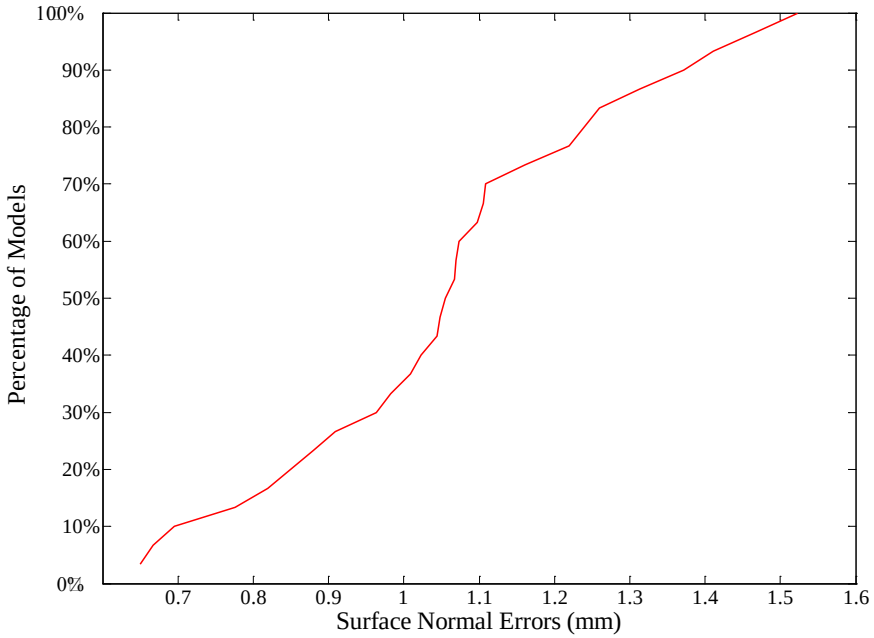



**Figure 13: Corrected RMS Surface Errors with Errors in All Members**

## 5.2 Analysis of Errors Due to Mesh Tessellation

As stated earlier, the reflector is not actually a sparse paraboloid but has a rectangular tessellation pattern as seen in Figure 14. The left of the figure shows the size of an individual panel and the right shows one bay of an arm, covered by six panels. Previous work by Greschik estimated the error due to tessellation would be less than 6 mm [7].

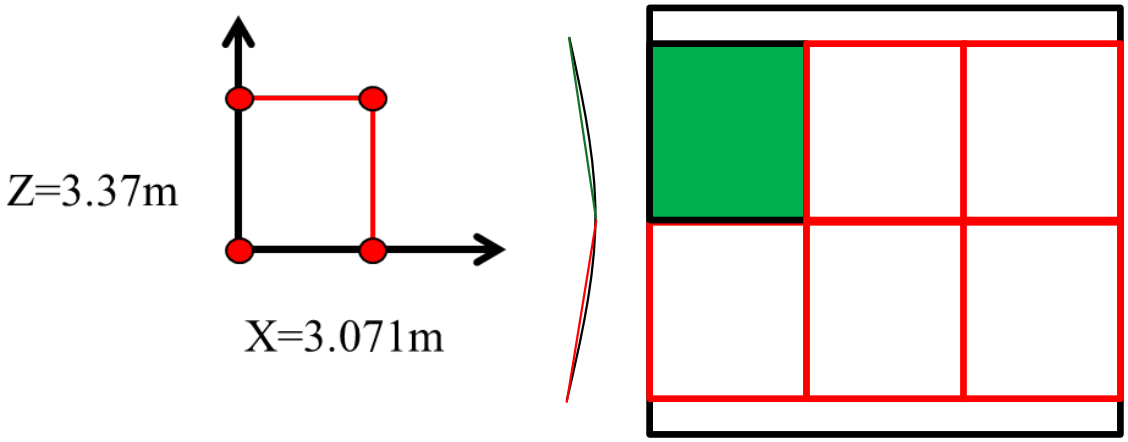



**Figure 14: Rectangular Mesh Panel**

The mesh tessellation error is caused by the panels differing from the ideal paraboloid described by Equation 18. A first analysis was performed using one of the six mesh panels in the first bay off the central truss, seen in green in Figure 14. On the left is the size of one panel, followed by a view down the length of an arm. The right of Figure 14 shows a top-down view of one reflector bay. The cross-section is seen in Figure 15 with the corners of the rectangular pattern incident with the ideal paraboloid. Using three corners of the rectangle, a plane is created with a comparison node every 10 cm.

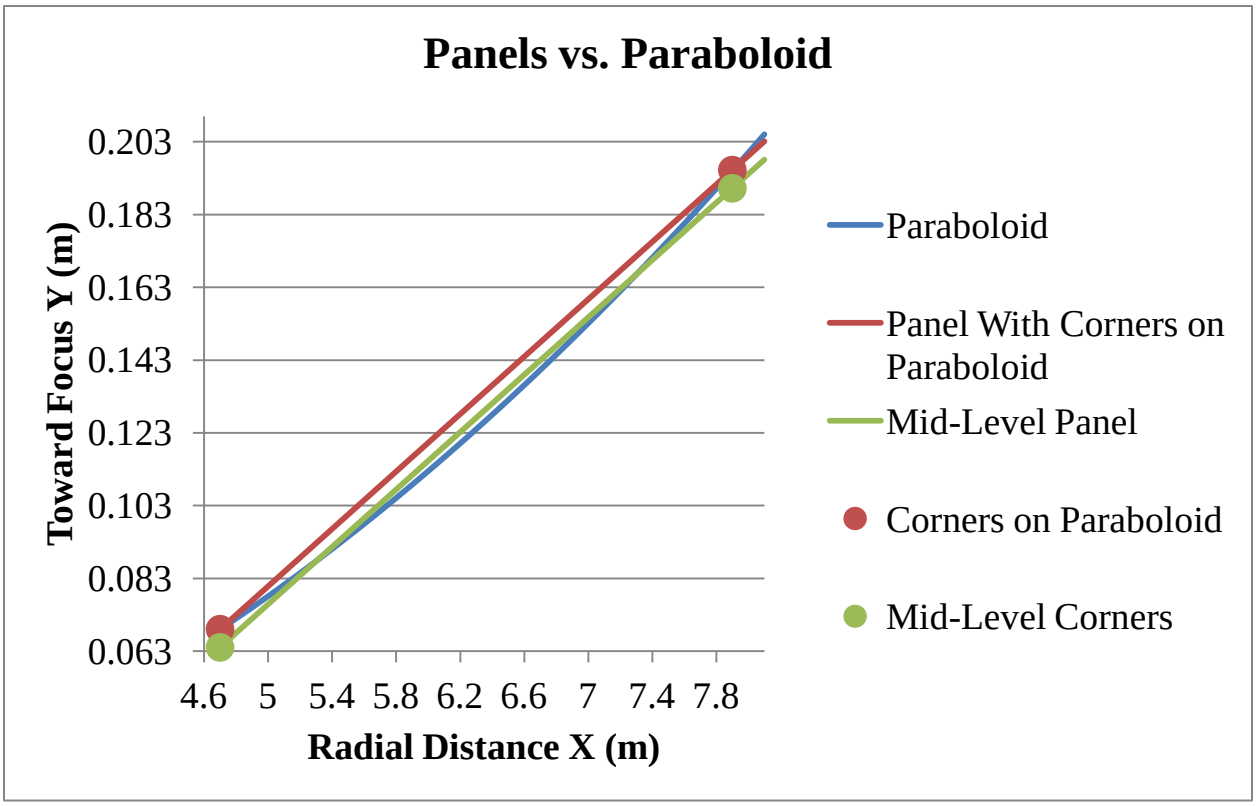



**Figure 15: Cross-Section View of Mesh Panels and Paraboloid**

The resulting 1,000+ comparison points are analyzed using the same method of Lagrange multiples to determine the $\varepsilon_{RMS}$. This process yields a RMS value of 10.61 mm for the error. Unsurprisingly this error is large as when calculating the RMS value, each error is squared. To minimize the $\varepsilon_{RMS}$ the maximum errors should be minimized. Lowering each panel so the center of the rectangle is mid-way to the ideal paraboloid yields a much more realistic error of 6.12 mm. A comparison of the surface normal errors at each point is shown in Figure 16.

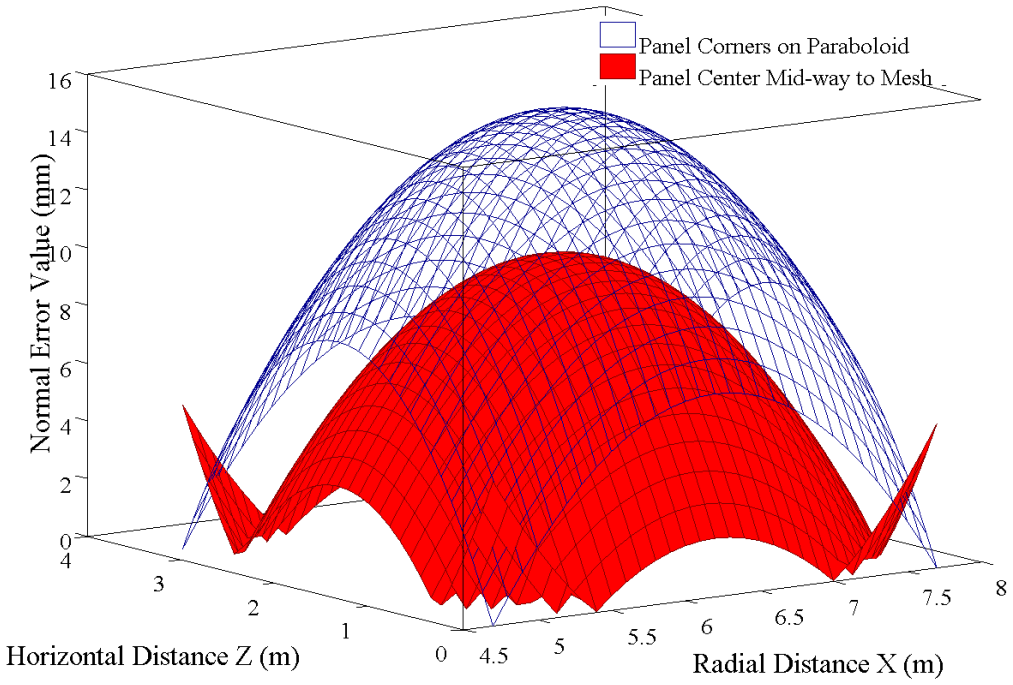
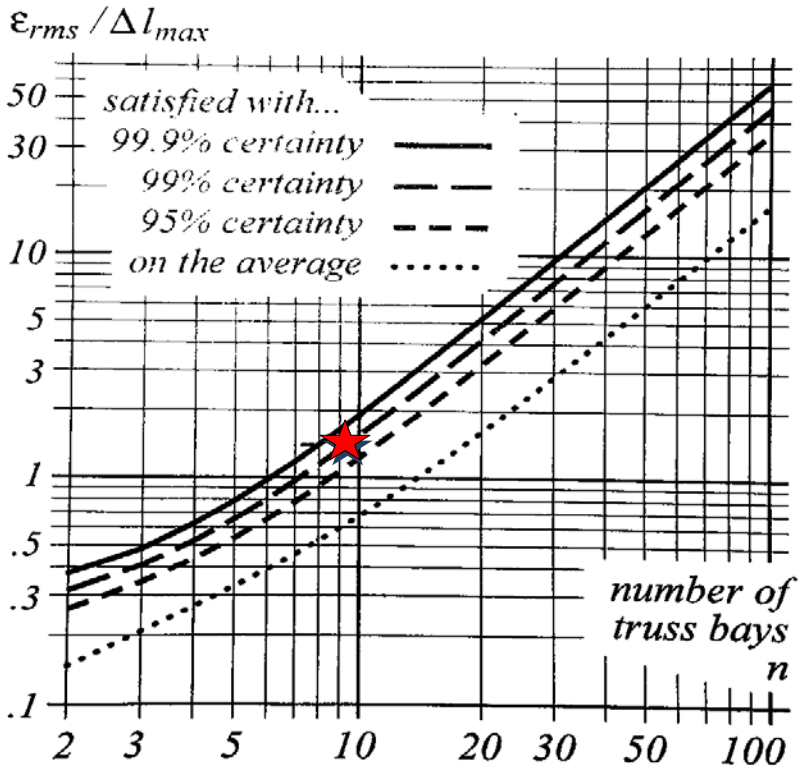



**Figure 16: Surface Normal Errors by Location**

### 5.3 Comparison of Cross-Shaped Reflector to References

Comparison of the cross-shaped reflector to the work in Reference [11] is fairly straight forward, despite the differing geometries. Each arm of the reflector is treated as a strip antenna with 8 bays, as they each extend from a central truss that in this analysis does not shift from the ideal. The RMS error from Reference [11] needs to be doubled, as they use half the surface normal offset instead of twice for a reflector. Using Figure 17 returns an estimated surface RMS of 0.2962 mm[†], within 16% of the 3σ value achieved here.

[†] $\Delta l_{max}$ = .1058 mm,



[11]

**Figure 17: $\varepsilon_{RMS}$ vs. Number of Bays in a Truss**

A second comparison produces an estimate that is nearly as close as the first, but in the other direction. Using Equation 7, 8 and 9 yields an estimate of 4.8870 mm for the average surface error, Equation 27. As the cross-shaped truss does not have a rim, removing the last term under the radical reduces the estimate to 3.6286 mm. This is within ~100% of the 3σ modeled value.

$$\frac{\varepsilon_{RMS}}{D} = \sigma_{\varepsilon}\sqrt{\frac{D}{l_b}\left(0.060\frac{l_b}{H}\right)^2 + \left(\frac{1}{2}\sqrt{\frac{l_b}{D}}\left(\frac{H}{l_b}+\frac{1}{4}*\frac{l_b}{H}\right)\right)^2 + \left(0.177\left(\frac{D}{L}+\frac{L}{D}\right)\right)^2} \quad \textbf{(27)}$$

where:
$\varepsilon_{RMS}$ = RMS surface error from truss errors
$\sigma_{\varepsilon}$ = Standard deviation of the unit length error for truss elements (4.932E-5 m)
$l_b$ = Radial length of a truss bay (9.2126 m)
$D$ = Reflector diameter (150 m)
$H$ = Height of truss bay (4.7 m)
$L$ = Total Length of Truss Arm (75 m)

Lastly a look at the expected error in the mesh surface because it is a series of tessellated squares. Per Equation 6, the error in the surface is a function of the size of the rectangles, as well as the overall dimensions of the antenna reflector. With the properties for the cross-shaped reflector, Equation 28, an average surface RMS error of 3.434 mm is achieved. This value is 56% of the value calculated for the sparse aperture tessellation pattern. The mesh tessellation error found by Greschik for the sparse aperture structure is within 2% of value calculated in this paper.

$$\varepsilon_{RMS} = \left(\frac{a}{D*7.33}*\left[1+\left(\frac{b}{a}\right)^4\right]^{1/4}\right)^2 * \frac{D^2}{F} \quad \textbf{(28)}$$

where:
$a$ = Larger rectangular dimension of mesh facet (3.37 m)
$D$ = Reflector diameter (150 m)
$F$ = Focal length (80 m)
$b$ = Shorter rectangular dimension of mesh facet (3.071 m)

Thus far the modeling performed here is in line with the accuracies achieved in published research on the topic. The values for the member-induced surface error in the cross-shaped sparse aperture antenna fall between the two estimates. Mesh tessellation error agrees with an earlier estimate and is within 44% of the values found for a filled-aperture spherical reflector.

### 5.4 Effect of Errors in Individual Truss Members

Running simulations consistent with the previous models on individual truss members will offer insight into which members have the most impact on the shape of the reflector surface. As the top longerons are connected directly to the mesh stanchions, it is likely they will have the largest effect on the mesh geometry. As shown in Figure 18 and Table 9, the errors in the top longerons do have the greatest contribution to the overall surface error, as seen when all members have up to $10^{-5}$ proportional errors in them.

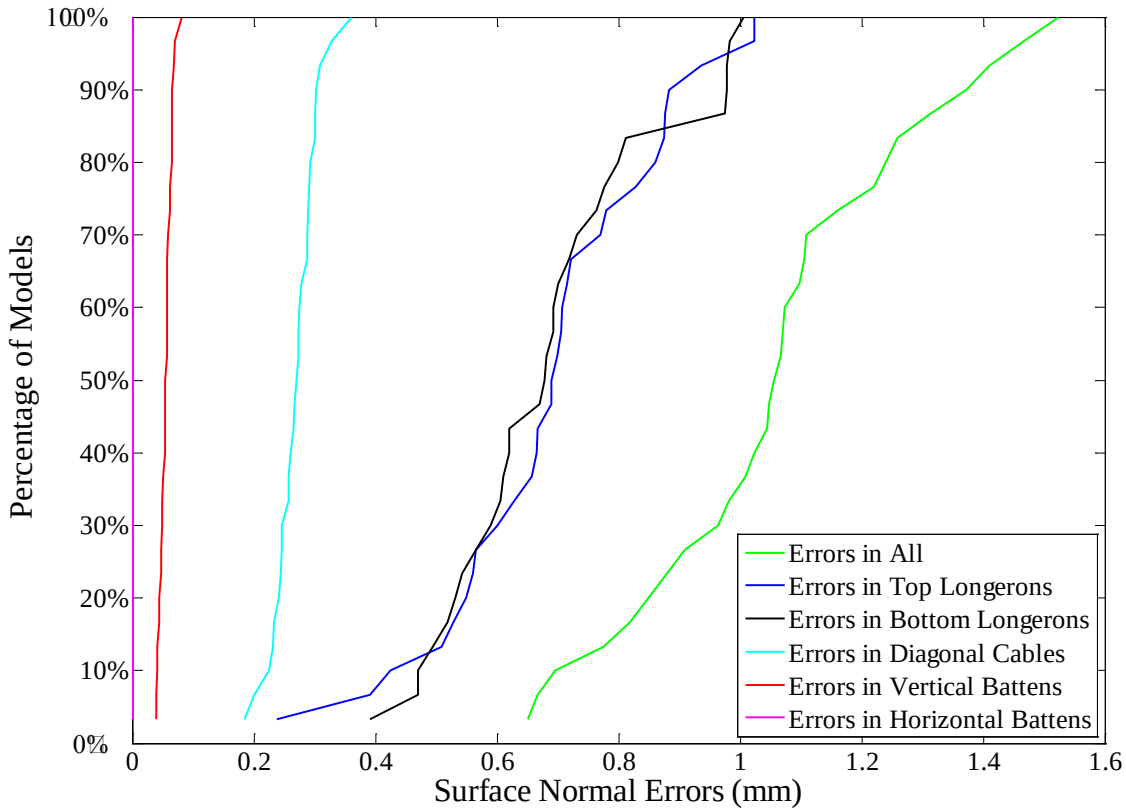



**Figure 18: Corrected RMS Surface Errors by Member Type**

The bottom longerons are longer, in general, by almost 3% than the top longerons. Thus bottom longerons have possible errors that are also 3% larger than the top longerons. Despite this, they have less of an impact on surface errors than the top longerons. As seen in Figure 18 and Table 9, the bottom longerons produce errors that are largely in line with the top longerons, within 3% for both the mean and 2σ surface error values. The main difference being that the top longerons produce a wider range of surface errors, but more clustered towards the higher end of their range.

Errors produced by the diagonal cables, seen in Figure 18, are much smaller than errors caused by the longerons. The mean surface error induced is less than half that produced by the longerons, and the $2\sigma$ error is only one third what the longerons produce. Although the diagonal cables are longer, and can have longer errors, there are opposing tension members, so the shape of the truss is not greatly changed when perturbed.

Errors in the shortest elements, the vertical battens shown in Figure 18, follow a predictable pattern. The mean surface error produced is on par with the maximum error allowed in the member, 0.047 mm. Errors caused by the vertical battens, while small, are almost exactly in the normal direction, Figure 19, which will cause a local error in the surface. Unlike the longerons or diagonal cables, these errors do not propagate to distal members, as a series of disparate length members in the radial direction will.

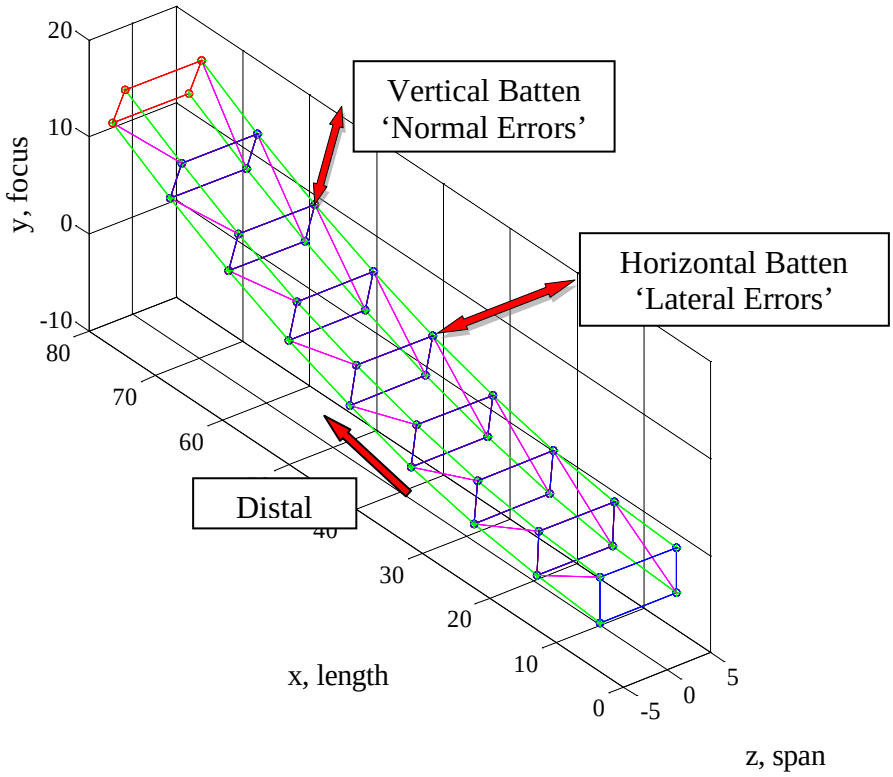



**Figure 19: Batten Error Directions**

As suspected, Figure 18 shows errors in the horizontal battens have the least contribution to geometric error of any member. These errors largely move the surface in a nearly lateral direction, Figure 19, which does not add to errors in the surface normal. These errors are also more localized, similarly to the vertical battens, which is evident in the tight cluster of surface error values. These errors are negligible to the overall surface error.

In summary, the contribution of each type of member error to the overall surface error is clearly visible. The largest errors occur, of course, when all members are allowed to deviate. The second largest effect is by errors in the longerons, the greatest associated

with errors in the top longerons. The next most important source of errors is from the diagonal cables differing in length, followed by the vertical battens. The horizontal battens have almost no effect, producing errors mostly in the radial direction. Comparing this to the uncorrected errors in Table 9 you will see the trends are maintained, but the underlying bias has been removed.

**Table 9: Results of Errors in Truss Members**

| Member with Errors ($10^{-5}$) | $\varepsilon_{RMS}$ Observed Uncorrected (mm) | | | $\varepsilon_{RMS}$ Observed Corrected (mm) | | |
|---|---|---|---|---|---|---|
| | **-2σ** | **Mean** | **+2σ** | **-2σ** | **Mean** | **+2σ** |
| **All** | 3.7545 | 4.2540 | 4.7536 | 0.6159 | 1.0619 | 1.5079 |
| **Top Longerons** | 3.7742 | 4.1646 | 4.5549 | 0.3379 | 0.6924 | 1.0469 |
| **Bottom Longerons** | 3.9103 | 4.2054 | 4.5005 | 0.3575 | 0.6892 | 1.0208 |
| **Diagonal Cable** | 4.0148 | 4.1093 | 4.2038 | 0.1979 | 0.2694 | 0.3410 |
| **Vertical Batten** | 4.0766 | 4.0971 | 4.1177 | 0.0355 | 0.0553 | 0.0752 |
| **Horizontal Batten** | 4.0956 | 4.0958 | 4.0959 | 0.0009 | 0.0012 | 0.0014 |

It is clear from Table 9 that the underlying truss model, labeled until this point as "ideal" needs to be corrected. The most likely source of errors has to do with how the tension cables are treated in the FEA model. Each cable is shortened to achieve a desired tension in the ideal geometry through a temperature change applied to the model. The code currently does not allow for cables distal to the central truss to change the proportion they shorten. This induces differing tensions based on the length of the diagonal cable, and likely twists the bays from ideal.

Up to this point all of the truss errors have been limited to a maximum proportional length error of $10^{-5}$. Relaxing the max proportional error to $10^{-4}$ increases the $\varepsilon_{RMS}$ by nearly a factor of nine. Comparing the $10^{-5}$ and $10^{-4}$ errors in Figure 20 shows a nearly linear proportionality in this regime.

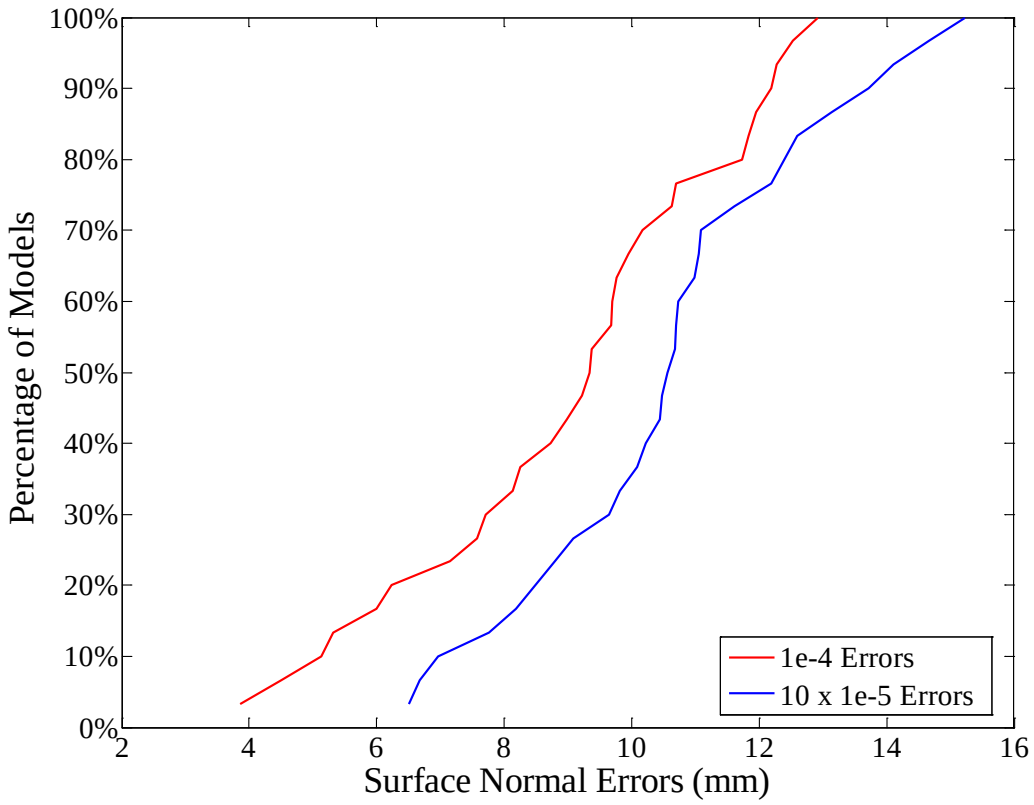



**Figure 20: RMS Error with Greater Variation**

### 5.5 Summary

As stated at the beginning of this thesis, normal surface deviation should be held below an RMS of 15 mm. High precision machining can obtain an error ratio of $10^{-5}$, which in this analysis yields a surface error of <1.53 mm, and is well within the required RMS. These values agree with published estimates of surface accuracy achievable in similar configurations. The ideal truss model needs to be improved to remove the surface error inherent in the model of ~4 mm. This is likely due to the way that tension cables are handled in the FEA model.

# VI. Conclusions and Recommendations

This research has examined the feasibility of using a very large, sparse aperture, cross-shaped, parabolic antenna for L-band communication. As an offshoot of previous work, it began with the dimensions for a flat, cross-shaped, deployable reflector and truss system. These types of antennas can be very useful in future communication systems, as they are deployable from current launch vehicles, and have an immense collection area, allowing for more robust and faster links.

Due to the uniquely large size of the space structure, an investigation of the intended operational environment at GEO was explored. To begin with, an annual $\Delta V$ budget for station keeping was developed to counter drift from the Sun, Moon and $J_{22}$ effects which is in line with other satellites at GEO. Next, the environmental torques due to solar pressure, gravity gradient and magnetic moment were examined. The torques were shown to be much larger than for a 50 m filled aperture antenna, but not significant enough to be of concern at this stage in development.

The geometry for the parabolic truss was established next, and the work herein followed the methodologies of published research in the field of large filled aperture and strip antennas, and extended it to sparse aperture parabolic reflectors. Different sources of error were introduced into the structure, and a method to solve for the effects on overall reflector surface accuracy was developed. These building blocks can easily be adapted to other parabolic reflectors, and with only minor modifications to other geometries.

## 6.1 Conclusions of Research

Despite the enormous size of the antenna reflector, and tight Diameter/$\varepsilon_{RMS}$ ratio of 10,000 required for L-band communication, the system seems feasible. The work performed for this thesis pulled on established techniques for analyzing full aperture, and strip, mesh antennas. The values achieved for truss induced surface error are in line with estimates from these techniques. With the mesh reflector and truss largely defined, there is still quite a bit of room left in the error budget seen in Table 10.

**Table 10: Antenna Surface Normal Error Budget**

| **Source** | **Estimated $\varepsilon_{RMS}$ (mm)** |
|---|---|
| Errors in Truss members | -1.5 |
| Errors in Surface Tessellation | -6.1 |
| **Reserve** | **7.4** |

Allowing for errors in the feed horn assembly similar to the rest of the truss, they should be on the order of 1 mm. Initial proposals for a stiffener system have cables running between the arms, which would produce a twisting motion, and should be largely neutral to surface normal errors. There are also cables running toward the feed horn, which would have an almost exclusively normal error, so should be considered carefully.

## 6.2 Significance of Research

This thesis drew on work for large space-based mesh antennas, but is the only instance seen to date using a sparse aperture parabolic antenna. Different contributions to error from the underlying truss versus a filled aperture pretensioned truss required a different method of resolving the errors in the mesh. This involved running FEA on models of the underlying truss with random errors in the members.

This approach was similar to that demonstrated with the flat strip antenna but the solution for the error normal could not be a simple integration. The parabolic shape required a method that would provide the distance between the error point of the mesh and a line normal to the ideal parabolic surface. Using the method of Lagrange multiples allowed an analytical solution to any error points found.

The methods and computer code developed are adaptable to other configurations, and with minor adjustments can give very quick estimates of errors in those structures. Further interpolation between the nodes found in this research will greatly increase the fidelity of the model. This design, if brought to completion, will allow the launch of much larger antennas than are currently in orbit using contemporary launch vehicles.

### 6.3 Recommendations for Future Research

The results achieved thus far point to two avenues of continuing research required to validate the very-large, sparse-aperture antenna concept. The first is a dynamic analysis of the antenna during operation, and the second involves further research into the stowage and deployment of the antenna.

The more pressing research requires completing the bulk design for the remainder of the antenna so a dynamic analysis can be performed. This mainly involves the structure to support the antenna feed horn and the stiffener system to stabilize it all. Due to the materials of and gossamer nature of the structure, unequal solar heating is unlikely to be an issue, but the settling time after pointing the antenna may limit its operational suitability.

Once the antenna is shown to be operationally usable, the concern is how to get it into orbit. Obviously the antenna needs to be stored and deployable in a predictable manner for orbit insertion. Of primary concern is maintaining stability during deployment to avoid damage to the antenna or bus.

## Appendix A – MATLAB® Code

### Guide

1. Load “Thesis_Data.m” file, it contains all of the variables required to run the code below. Things are greatly sped up by not re-solving for these values.

2. Run “NASTRAN_BulkProcesses_2b.m” this function runs NASTRAN static FEA and outputs the geometries of the node points to a MATLAB data file.

    a. Line 13 specifies the total number of models to test.

    b. Calls “Error_Points_rand_XXX”, must specify in “NASTRAN_BulkProcesses_2b.m” which file to run, depending on the members you want errors in.

        i. This function provides the initial locations of each node in the model.

        ii. Must specify in “Error_Points_Rand_XXX” on line 6 the maximum size of error you are allowing.

    c. Calls “BoxFind_3.m” which takes the initial central truss node locations and all member lengths to solve for the location of the un-tensioned nodes.

    d. Calls “ToDATFile_2” which processes the points from “Error_Points…” function and builds the NASTRAN cards

        i. All material properties are contained in this file for FEA.

        ii. Solution for diagonals to pre-tension truss is found here.

    e. Uses “NASTRAN_Header_2c.blk” which is the start of each FEA bulk data file.

3. Run “sigma_root_mean_square.m” this function uses the “Data.m” file from “NASTRAN_BulkProcesses_2b.m” and produces the error lengths for the nodes in each model and corrects to remove the error from the non-ideal truss.

    a. Must first run this file with the no-error model to find the corrected values in "ideal_norm_errors” variable.

    b. Calls “Surface_Normal_Determination.m” which solves the Lagrangians for each error point to an ideal paraboloid.

    c. Plots the RMS surface normal error for all of the models.

4. Run “Mesh_tessellation_error.m” this function calculates the error due to mesh tessellation using “Surface_Normal_Determination.m” without truss offset.

**NASTRAN_BulkProcesses_2b.m, runs FEA trials and compiles data:**

```
% Take points, generate DAT file and then have NASTRAN run analysis
% file name of blk file
FileName = @(x) ['TestOutput_20131216_k' num2str(x,'%03u') '.blk'];

DirPath =
'L:\Research\LargeSpaceStructure\ManufacturingAccuracy\FEMAP\Sandbox3\'
;
% DirPath = [cd, '\']; % use current directory, or some relation to it

RunList=1:30; %number of trials
Flagz.ShowResults=true;
%% generate DAT file

for Run=RunList;

[Pnts_all]=Error_Points_rand_all();

ToDATFile_2(Pnts_all,FileName(Run));
end

%% Run analysis
for Run =RunList
    FNow=FileName(Run);
[a,b]=system(['cmd /c ""R:\ENY
Applications\MSC.Software\MDNastranR3\bin\mdnastran"" ',...
    ' ',DirPath ,FNow,...
    ' out=',DirPath , FNow(1:(end-4)), '_out'],'-echo');
end


%% Read results
Data = cell(0,1);
if Flagz.ShowResults; figure(428);clf;end
for Run =RunList

% read starting point
Data_inn=zeros(0,4);
FNow=FileName(Run);
fido = fopen([DirPath , FNow]);

temp_line=fgetl(fido);temp_count=1;
while ~feof(fido)||temp_count>1e5
if (length(temp_line)>4)&&strcmpi(temp_line(1:4),'GRID')
    Data_inn(end+1,:)=str2num([temp_line((1:8)+8*(2-
1));temp_line((1:8)+8*(4-1));temp_line((1:8)+8*(5-
1));temp_line((1:8)+8*(6-1))]).';
end
temp_line=fgetl(fido);
temp_count=temp_count+1;
end
fclose(fido);
```

```
% read output
Data_out=zeros(0,4);
FNow=FileName(Run);
fido = fopen([DirPath , FNow(1:(end-4)), '_out.f06']);

temp_line=fgetl(fido);temp_count=1;
while ~feof(fido)||temp_count>1e5
if strcmpi(temp_line,'                                                D I
S P L A C E M E N T   V E C T O R')
    temp_line=fgetl(fido);% blank line
    temp_line=fgetl(fido);% headers
    temp_flag=true;
    while temp_flag
    temp_line=fgetl(fido);% Data
    temp_G = textscan(temp_line,'%f G %f %f %f %*f %*f %*f');
    if length([temp_G{:}])<4
        temp_flag = false;
    else
    Data_out(end+1,:)=cell2mat(temp_G);
    end
    end
end
temp_line=fgetl(fido);
temp_count=temp_count+1;
end
fclose(fido);

[dummy,temp_s1]=sort(Data_inn(:,1));
[dummy,temp_s2]=sort(Data_out(:,1));
Data_inns=Data_inn(temp_s1,:);
Data_outs=Data_out(temp_s2,:);
if sum(abs(Data_inns(:,1)-Data_outs(:,1)))
warning('Doesn`t look like you got the points to match up...Dope!');
end
Data{Run}=Data_inns+Data_outs;
if Flagz.ShowResults;
plot3(Data{Run}(:,4),Data{Run}(:,2),Data{Run}(:,3),'.');end
end
if Flagz.ShowResults;
daspect([1,1,1]);xlabel('z');ylabel('x');zlabel('y');end

%% save results

save(['Data_' datestr(now,'yyyy_mm_dd__HH_MM') '.mat'],'Data');
```

**Error_Points_rand_all().m, node location calculator with random errors included:**

```
function [Pnts_all]=Error_Points_rand_all()

global x y xbot ybot lbot l d w D_1 D_2 D_3 D_4 D_5 D_6 D_7 D_8
Pnts_all...
    rightarm leftarm backarm frontarm

maxerror=-.00001; % proportional error allowed in each truss member

% Right arm
Longs=cell(8,1);
LR1=[(1+maxerror - 2*maxerror*rand())*l;(1+maxerror -
2*maxerror*rand())*lbot(1);(1+maxerror -
2*maxerror*rand())*l;(1+maxerror - 2*maxerror*rand())*lbot(1)];
LR2=[(1+maxerror - 2*maxerror*rand())*l;(1+maxerror -
2*maxerror*rand())*lbot(2);(1+maxerror -
2*maxerror*rand())*l;(1+maxerror - 2*maxerror*rand())*lbot(2)];
LR3=[(1+maxerror - 2*maxerror*rand())*l;(1+maxerror -
2*maxerror*rand())*lbot(3);(1+maxerror -
2*maxerror*rand())*l;(1+maxerror - 2*maxerror*rand())*lbot(3)];
LR4=[(1+maxerror - 2*maxerror*rand())*l;(1+maxerror -
2*maxerror*rand())*lbot(4);(1+maxerror -
2*maxerror*rand())*l;(1+maxerror - 2*maxerror*rand())*lbot(4)];
LR5=[(1+maxerror - 2*maxerror*rand())*l;(1+maxerror -
2*maxerror*rand())*lbot(5);(1+maxerror -
2*maxerror*rand())*l;(1+maxerror - 2*maxerror*rand())*lbot(5)];
LR6=[(1+maxerror - 2*maxerror*rand())*l;(1+maxerror -
2*maxerror*rand())*lbot(6);(1+maxerror -
2*maxerror*rand())*l;(1+maxerror - 2*maxerror*rand())*lbot(6)];
LR7=[(1+maxerror - 2*maxerror*rand())*l;(1+maxerror -
2*maxerror*rand())*lbot(7);(1+maxerror -
2*maxerror*rand())*l;(1+maxerror - 2*maxerror*rand())*lbot(7)];
LR8=[(1+maxerror - 2*maxerror*rand())*l;(1+maxerror -
2*maxerror*rand())*lbot(8);(1+maxerror -
2*maxerror*rand())*l;(1+maxerror - 2*maxerror*rand())*lbot(8)];
Longs(1)={LR1};Longs(2)={LR2};Longs(3)={LR3};Longs(4)={LR4};Longs(5)={L
R5};Longs(6)={LR6};Longs(7)={LR7};Longs(8)={LR8};

Battens=cell(9,1);
BR1=[(1+maxerror - 2*maxerror*rand())*d;(1+maxerror -
2*maxerror*rand())*w;(1+maxerror - 2*maxerror*rand())*w;(1+maxerror -
2*maxerror*rand())*d];
BR2=[(1+maxerror - 2*maxerror*rand())*d;(1+maxerror -
2*maxerror*rand())*w;(1+maxerror - 2*maxerror*rand())*w;(1+maxerror -
2*maxerror*rand())*d];
BR3=[(1+maxerror - 2*maxerror*rand())*d;(1+maxerror -
2*maxerror*rand())*w;(1+maxerror - 2*maxerror*rand())*w;(1+maxerror -
2*maxerror*rand())*d];
BR4=[(1+maxerror - 2*maxerror*rand())*d;(1+maxerror -
2*maxerror*rand())*w;(1+maxerror - 2*maxerror*rand())*w;(1+maxerror -
2*maxerror*rand())*d];
```

```
BR5=[(1+maxerror - 2*maxerror*rand())*d;(1+maxerror -
2*maxerror*rand())*w;(1+maxerror - 2*maxerror*rand())*w;(1+maxerror -
2*maxerror*rand())*d];
BR6=[(1+maxerror - 2*maxerror*rand())*d;(1+maxerror -
2*maxerror*rand())*w;(1+maxerror - 2*maxerror*rand())*w;(1+maxerror -
2*maxerror*rand())*d];
BR7=[(1+maxerror - 2*maxerror*rand())*d;(1+maxerror -
2*maxerror*rand())*w;(1+maxerror - 2*maxerror*rand())*w;(1+maxerror -
2*maxerror*rand())*d];
BR8=[(1+maxerror - 2*maxerror*rand())*d;(1+maxerror -
2*maxerror*rand())*w;(1+maxerror - 2*maxerror*rand())*w;(1+maxerror -
2*maxerror*rand())*d];
BR9=[(1+maxerror - 2*maxerror*rand())*d;(1+maxerror -
2*maxerror*rand())*w;(1+maxerror - 2*maxerror*rand())*w;(1+maxerror -
2*maxerror*rand())*d];
Battens(1)={BR1};Battens(2)={BR2};Battens(3)={BR3};Battens(4)={BR4};Bat
tens(5)={BR5};Battens(6)={BR6};Battens(7)={BR7};Battens(8)={BR8};Batten
s(9)={BR9};

D0=cell(8,1);
D0(1)={[[(1+maxerror - 2*maxerror*rand()),0;0,(1+maxerror -
2*maxerror*rand())]*D_1']'};...
    D0(2)={[[(1+maxerror - 2*maxerror*rand()),0;0,(1+maxerror -
2*maxerror*rand())]*D_2']'};...
    D0(3)={[[(1+maxerror - 2*maxerror*rand()),0;0,(1+maxerror -
2*maxerror*rand())]*D_3']'};...
    D0(4)={[[(1+maxerror - 2*maxerror*rand()),0;0,(1+maxerror -
2*maxerror*rand())]*D_4']'};...
    D0(5)={[[(1+maxerror - 2*maxerror*rand()),0;0,(1+maxerror -
2*maxerror*rand())]*D_5']'};...
    D0(6)={[[(1+maxerror - 2*maxerror*rand()),0;0,(1+maxerror -
2*maxerror*rand())]*D_6']'};...
    D0(7)={[[(1+maxerror - 2*maxerror*rand()),0;0,(1+maxerror -
2*maxerror*rand())]*D_7']'};...
    D0(8)={[[(1+maxerror - 2*maxerror*rand()),0;0,(1+maxerror -
2*maxerror*rand())]*D_8']'};



Pnts_int=[x(2),    y(2),     w/2;...
          xbot(2), ybot(2), w/2;...
          x(2),    y(2),     -w/2;...
          xbot(2), ybot(2), -w/2];

Pnts_all=cell(4,1);
rightarm=cell(4,3,9);

for redsox=1:length(lbot);
    for celtics=1:4
        rightarm{celtics,1,redsox}=Pnts_int(celtics,1);
        rightarm{celtics,2,redsox}=Pnts_int(celtics,2);
        rightarm{celtics,3,redsox}=Pnts_int(celtics,3);
    end
```

```
[Pnts_outer,Residual]=BoxFind_3(Pnts_int,Longs{redsox},Battens{redsox},
D0{redsox});
    Pnts_int=Pnts_outer;
end
for celtics=1:4
        rightarm{celtics,1,redsox+1}=Pnts_outer(celtics,1);
        rightarm{celtics,2,redsox+1}=Pnts_outer(celtics,2);
        rightarm{celtics,3,redsox+1}=Pnts_outer(celtics,3);
end
rightarm=cell2mat(rightarm);
Pnts_all{1}=rightarm;

%% Back arm
Longs=cell(8,1);
LR1=[(1+maxerror - 2*maxerror*rand())*l;(1+maxerror -
2*maxerror*rand())*lbot(1);(1+maxerror -
2*maxerror*rand())*l;(1+maxerror - 2*maxerror*rand())*lbot(1)];
LR2=[(1+maxerror - 2*maxerror*rand())*l;(1+maxerror -
2*maxerror*rand())*lbot(2);(1+maxerror -
2*maxerror*rand())*l;(1+maxerror - 2*maxerror*rand())*lbot(2)];
LR3=[(1+maxerror - 2*maxerror*rand())*l;(1+maxerror -
2*maxerror*rand())*lbot(3);(1+maxerror -
2*maxerror*rand())*l;(1+maxerror - 2*maxerror*rand())*lbot(3)];
LR4=[(1+maxerror - 2*maxerror*rand())*l;(1+maxerror -
2*maxerror*rand())*lbot(4);(1+maxerror -
2*maxerror*rand())*l;(1+maxerror - 2*maxerror*rand())*lbot(4)];
LR5=[(1+maxerror - 2*maxerror*rand())*l;(1+maxerror -
2*maxerror*rand())*lbot(5);(1+maxerror -
2*maxerror*rand())*l;(1+maxerror - 2*maxerror*rand())*lbot(5)];
LR6=[(1+maxerror - 2*maxerror*rand())*l;(1+maxerror -
2*maxerror*rand())*lbot(6);(1+maxerror -
2*maxerror*rand())*l;(1+maxerror - 2*maxerror*rand())*lbot(6)];
LR7=[(1+maxerror - 2*maxerror*rand())*l;(1+maxerror -
2*maxerror*rand())*lbot(7);(1+maxerror -
2*maxerror*rand())*l;(1+maxerror - 2*maxerror*rand())*lbot(7)];
LR8=[(1+maxerror - 2*maxerror*rand())*l;(1+maxerror -
2*maxerror*rand())*lbot(8);(1+maxerror -
2*maxerror*rand())*l;(1+maxerror - 2*maxerror*rand())*lbot(8)];
Longs(1)={LR1};Longs(2)={LR2};Longs(3)={LR3};Longs(4)={LR4};Longs(5)={L
R5};Longs(6)={LR6};Longs(7)={LR7};Longs(8)={LR8};

Battens=cell(9,1);
BR1=[(1+maxerror - 2*maxerror*rand())*d;(1+maxerror -
2*maxerror*rand())*w;(1+maxerror - 2*maxerror*rand())*w;(1+maxerror -
2*maxerror*rand())*d];
BR2=[(1+maxerror - 2*maxerror*rand())*d;(1+maxerror -
2*maxerror*rand())*w;(1+maxerror - 2*maxerror*rand())*w;(1+maxerror -
2*maxerror*rand())*d];
BR3=[(1+maxerror - 2*maxerror*rand())*d;(1+maxerror -
2*maxerror*rand())*w;(1+maxerror - 2*maxerror*rand())*w;(1+maxerror -
2*maxerror*rand())*d];
BR4=[(1+maxerror - 2*maxerror*rand())*d;(1+maxerror -
2*maxerror*rand())*w;(1+maxerror - 2*maxerror*rand())*w;(1+maxerror -
2*maxerror*rand())*d];
```

```
BR5=[(1+maxerror - 2*maxerror*rand())*d;(1+maxerror -
2*maxerror*rand())*w;(1+maxerror - 2*maxerror*rand())*w;(1+maxerror -
2*maxerror*rand())*d];
BR6=[(1+maxerror - 2*maxerror*rand())*d;(1+maxerror -
2*maxerror*rand())*w;(1+maxerror - 2*maxerror*rand())*w;(1+maxerror -
2*maxerror*rand())*d];
BR7=[(1+maxerror - 2*maxerror*rand())*d;(1+maxerror -
2*maxerror*rand())*w;(1+maxerror - 2*maxerror*rand())*w;(1+maxerror -
2*maxerror*rand())*d];
BR8=[(1+maxerror - 2*maxerror*rand())*d;(1+maxerror -
2*maxerror*rand())*w;(1+maxerror - 2*maxerror*rand())*w;(1+maxerror -
2*maxerror*rand())*d];
BR9=[(1+maxerror - 2*maxerror*rand())*d;(1+maxerror -
2*maxerror*rand())*w;(1+maxerror - 2*maxerror*rand())*w;(1+maxerror -
2*maxerror*rand())*d];
Battens(1)={BR1};Battens(2)={BR2};Battens(3)={BR3};Battens(4)={BR4};Bat
tens(5)={BR5};Battens(6)={BR6};Battens(7)={BR7};Battens(8)={BR8};Batten
s(9)={BR9};


D0=cell(9,1);
D0(1)={[[(1+maxerror - 2*maxerror*rand()),0;0,(1+maxerror -
2*maxerror*rand())]*D_1']'};...
    D0(2)={[[(1+maxerror - 2*maxerror*rand()),0;0,(1+maxerror -
2*maxerror*rand())]*D_2']'};...
    D0(3)={[[(1+maxerror - 2*maxerror*rand()),0;0,(1+maxerror -
2*maxerror*rand())]*D_3']'};...
    D0(4)={[[(1+maxerror - 2*maxerror*rand()),0;0,(1+maxerror -
2*maxerror*rand())]*D_4']'};...
    D0(5)={[[(1+maxerror - 2*maxerror*rand()),0;0,(1+maxerror -
2*maxerror*rand())]*D_5']'};...
    D0(6)={[[(1+maxerror - 2*maxerror*rand()),0;0,(1+maxerror -
2*maxerror*rand())]*D_6']'};...
    D0(7)={[[(1+maxerror - 2*maxerror*rand()),0;0,(1+maxerror -
2*maxerror*rand())]*D_7']'};...
    D0(8)={[[(1+maxerror - 2*maxerror*rand()),0;0,(1+maxerror -
2*maxerror*rand())]*D_8']'};

Pnts_int=[x(2),    y(2),     w/2;...
          xbot(2), ybot(2), w/2;...
          x(2),    y(2),     -w/2;...
          xbot(2), ybot(2), -w/2];

backarm=cell(4,3,9);

for redsox=1:length(lbot);
    for celtics=1:4
        Pnts_int1=Pnts_int*[0,0,1;0,1,0;-1,0,0];
        backarm{celtics,1,redsox}=Pnts_int1(celtics,1);
        backarm{celtics,2,redsox}=Pnts_int1(celtics,2);
        backarm{celtics,3,redsox}=Pnts_int1(celtics,3);
    end
```

```
[Pnts_outer,Residual]=BoxFind_3(Pnts_int,Longs{redsox},Battens{redsox},
D0{redsox});
    Pnts_int=Pnts_outer;
end
for celtics=1:4
    Pnts_outer1=Pnts_outer*[0,0,1;0,1,0;-1,0,0];
    backarm{celtics,1,redsox+1}=Pnts_outer1(celtics,1);
    backarm{celtics,2,redsox+1}=Pnts_outer1(celtics,2);
    backarm{celtics,3,redsox+1}=Pnts_outer1(celtics,3);
end
backarm=cell2mat(backarm);
Pnts_all{2}=backarm;

%% Left arm
Longs=cell(8,1);
LR1=[(1+maxerror - 2*maxerror*rand())*l;(1+maxerror -
2*maxerror*rand())*lbot(1);(1+maxerror -
2*maxerror*rand())*l;(1+maxerror - 2*maxerror*rand())*lbot(1)];
LR2=[(1+maxerror - 2*maxerror*rand())*l;(1+maxerror -
2*maxerror*rand())*lbot(2);(1+maxerror -
2*maxerror*rand())*l;(1+maxerror - 2*maxerror*rand())*lbot(2)];
LR3=[(1+maxerror - 2*maxerror*rand())*l;(1+maxerror -
2*maxerror*rand())*lbot(3);(1+maxerror -
2*maxerror*rand())*l;(1+maxerror - 2*maxerror*rand())*lbot(3)];
LR4=[(1+maxerror - 2*maxerror*rand())*l;(1+maxerror -
2*maxerror*rand())*lbot(4);(1+maxerror -
2*maxerror*rand())*l;(1+maxerror - 2*maxerror*rand())*lbot(4)];
LR5=[(1+maxerror - 2*maxerror*rand())*l;(1+maxerror -
2*maxerror*rand())*lbot(5);(1+maxerror -
2*maxerror*rand())*l;(1+maxerror - 2*maxerror*rand())*lbot(5)];
LR6=[(1+maxerror - 2*maxerror*rand())*l;(1+maxerror -
2*maxerror*rand())*lbot(6);(1+maxerror -
2*maxerror*rand())*l;(1+maxerror - 2*maxerror*rand())*lbot(6)];
LR7=[(1+maxerror - 2*maxerror*rand())*l;(1+maxerror -
2*maxerror*rand())*lbot(7);(1+maxerror -
2*maxerror*rand())*l;(1+maxerror - 2*maxerror*rand())*lbot(7)];
LR8=[(1+maxerror - 2*maxerror*rand())*l;(1+maxerror -
2*maxerror*rand())*lbot(8);(1+maxerror -
2*maxerror*rand())*l;(1+maxerror - 2*maxerror*rand())*lbot(8)];
Longs(1)={LR1};Longs(2)={LR2};Longs(3)={LR3};Longs(4)={LR4};Longs(5)={L
R5};Longs(6)={LR6};Longs(7)={LR7};Longs(8)={LR8};

Battens=cell(9,1);
BR1=[(1+maxerror - 2*maxerror*rand())*d;(1+maxerror -
2*maxerror*rand())*w;(1+maxerror - 2*maxerror*rand())*w;(1+maxerror -
2*maxerror*rand())*d];
BR2=[(1+maxerror - 2*maxerror*rand())*d;(1+maxerror -
2*maxerror*rand())*w;(1+maxerror - 2*maxerror*rand())*w;(1+maxerror -
2*maxerror*rand())*d];
BR3=[(1+maxerror - 2*maxerror*rand())*d;(1+maxerror -
2*maxerror*rand())*w;(1+maxerror - 2*maxerror*rand())*w;(1+maxerror -
2*maxerror*rand())*d];
```

```
BR4=[(1+maxerror - 2*maxerror*rand())*d;(1+maxerror -
2*maxerror*rand())*w;(1+maxerror - 2*maxerror*rand())*w;(1+maxerror -
2*maxerror*rand())*d];
BR5=[(1+maxerror - 2*maxerror*rand())*d;(1+maxerror -
2*maxerror*rand())*w;(1+maxerror - 2*maxerror*rand())*w;(1+maxerror -
2*maxerror*rand())*d];
BR6=[(1+maxerror - 2*maxerror*rand())*d;(1+maxerror -
2*maxerror*rand())*w;(1+maxerror - 2*maxerror*rand())*w;(1+maxerror -
2*maxerror*rand())*d];
BR7=[(1+maxerror - 2*maxerror*rand())*d;(1+maxerror -
2*maxerror*rand())*w;(1+maxerror - 2*maxerror*rand())*w;(1+maxerror -
2*maxerror*rand())*d];
BR8=[(1+maxerror - 2*maxerror*rand())*d;(1+maxerror -
2*maxerror*rand())*w;(1+maxerror - 2*maxerror*rand())*w;(1+maxerror -
2*maxerror*rand())*d];
BR9=[(1+maxerror - 2*maxerror*rand())*d;(1+maxerror -
2*maxerror*rand())*w;(1+maxerror - 2*maxerror*rand())*w;(1+maxerror -
2*maxerror*rand())*d];
Battens(1)={BR1};Battens(2)={BR2};Battens(3)={BR3};Battens(4)={BR4};Bat
tens(5)={BR5};Battens(6)={BR6};Battens(7)={BR7};Battens(8)={BR8};Batten
s(9)={BR9};

D0=cell(9,1);
D0(1)={[[(1+maxerror - 2*maxerror*rand()),0;0,(1+maxerror -
2*maxerror*rand())]*D_1']'};...
    D0(2)={[[(1+maxerror - 2*maxerror*rand()),0;0,(1+maxerror -
2*maxerror*rand())]*D_2']'};...
    D0(3)={[[(1+maxerror - 2*maxerror*rand()),0;0,(1+maxerror -
2*maxerror*rand())]*D_3']'};...
    D0(4)={[[(1+maxerror - 2*maxerror*rand()),0;0,(1+maxerror -
2*maxerror*rand())]*D_4']'};...
    D0(5)={[[(1+maxerror - 2*maxerror*rand()),0;0,(1+maxerror -
2*maxerror*rand())]*D_5']'};...
    D0(6)={[[(1+maxerror - 2*maxerror*rand()),0;0,(1+maxerror -
2*maxerror*rand())]*D_6']'};...
    D0(7)={[[(1+maxerror - 2*maxerror*rand()),0;0,(1+maxerror -
2*maxerror*rand())]*D_7']'};...
    D0(8)={[[(1+maxerror - 2*maxerror*rand()),0;0,(1+maxerror -
2*maxerror*rand())]*D_8']'};

Pnts_int=[x(2),    y(2),     w/2;...
          xbot(2), ybot(2), w/2;...
          x(2),    y(2),     -w/2;...
          xbot(2), ybot(2), -w/2];

leftarm=cell(4,3,9);

for redsox=1:length(lbot);
    for celtics=1:4
        Pnts_int1=Pnts_int*[-1,0,0;0,1,0;0,0,-1];
        leftarm{celtics,1,redsox}=Pnts_int1(celtics,1);
        leftarm{celtics,2,redsox}=Pnts_int1(celtics,2);
        leftarm{celtics,3,redsox}=Pnts_int1(celtics,3);
    end
```

```
[Pnts_outer,Residual]=BoxFind_3(Pnts_int,Longs{redsox},Battens{redsox},
D0{redsox});
    Pnts_int=Pnts_outer;
end
for celtics=1:4
    Pnts_outer1=Pnts_outer*[-1,0,0;0,1,0;0,0,-1];
    leftarm{celtics,1,redsox+1}=Pnts_outer1(celtics,1);
    leftarm{celtics,2,redsox+1}=Pnts_outer1(celtics,2);
    leftarm{celtics,3,redsox+1}=Pnts_outer1(celtics,3);
end
leftarm=cell2mat(leftarm);
Pnts_all{3}=leftarm;

%% Front arm
Longs=cell(8,1);
LR1=[(1+maxerror - 2*maxerror*rand())*l;(1+maxerror -
2*maxerror*rand())*lbot(1);(1+maxerror -
2*maxerror*rand())*l;(1+maxerror - 2*maxerror*rand())*lbot(1)];
LR2=[(1+maxerror - 2*maxerror*rand())*l;(1+maxerror -
2*maxerror*rand())*lbot(2);(1+maxerror -
2*maxerror*rand())*l;(1+maxerror - 2*maxerror*rand())*lbot(2)];
LR3=[(1+maxerror - 2*maxerror*rand())*l;(1+maxerror -
2*maxerror*rand())*lbot(3);(1+maxerror -
2*maxerror*rand())*l;(1+maxerror - 2*maxerror*rand())*lbot(3)];
LR4=[(1+maxerror - 2*maxerror*rand())*l;(1+maxerror -
2*maxerror*rand())*lbot(4);(1+maxerror -
2*maxerror*rand())*l;(1+maxerror - 2*maxerror*rand())*lbot(4)];
LR5=[(1+maxerror - 2*maxerror*rand())*l;(1+maxerror -
2*maxerror*rand())*lbot(5);(1+maxerror -
2*maxerror*rand())*l;(1+maxerror - 2*maxerror*rand())*lbot(5)];
LR6=[(1+maxerror - 2*maxerror*rand())*l;(1+maxerror -
2*maxerror*rand())*lbot(6);(1+maxerror -
2*maxerror*rand())*l;(1+maxerror - 2*maxerror*rand())*lbot(6)];
LR7=[(1+maxerror - 2*maxerror*rand())*l;(1+maxerror -
2*maxerror*rand())*lbot(7);(1+maxerror -
2*maxerror*rand())*l;(1+maxerror - 2*maxerror*rand())*lbot(7)];
LR8=[(1+maxerror - 2*maxerror*rand())*l;(1+maxerror -
2*maxerror*rand())*lbot(8);(1+maxerror -
2*maxerror*rand())*l;(1+maxerror - 2*maxerror*rand())*lbot(8)];
Longs(1)={LR1};Longs(2)={LR2};Longs(3)={LR3};Longs(4)={LR4};Longs(5)={L
R5};Longs(6)={LR6};Longs(7)={LR7};Longs(8)={LR8};

Battens=cell(9,1);
BR1=[(1+maxerror - 2*maxerror*rand())*d;(1+maxerror -
2*maxerror*rand())*w;(1+maxerror - 2*maxerror*rand())*w;(1+maxerror -
2*maxerror*rand())*d];
BR2=[(1+maxerror - 2*maxerror*rand())*d;(1+maxerror -
2*maxerror*rand())*w;(1+maxerror - 2*maxerror*rand())*w;(1+maxerror -
2*maxerror*rand())*d];
BR3=[(1+maxerror - 2*maxerror*rand())*d;(1+maxerror -
2*maxerror*rand())*w;(1+maxerror - 2*maxerror*rand())*w;(1+maxerror -
2*maxerror*rand())*d];
```

```
BR4=[(1+maxerror - 2*maxerror*rand())*d;(1+maxerror -
2*maxerror*rand())*w;(1+maxerror - 2*maxerror*rand())*w;(1+maxerror -
2*maxerror*rand())*d];
BR5=[(1+maxerror - 2*maxerror*rand())*d;(1+maxerror -
2*maxerror*rand())*w;(1+maxerror - 2*maxerror*rand())*w;(1+maxerror -
2*maxerror*rand())*d];
BR6=[(1+maxerror - 2*maxerror*rand())*d;(1+maxerror -
2*maxerror*rand())*w;(1+maxerror - 2*maxerror*rand())*w;(1+maxerror -
2*maxerror*rand())*d];
BR7=[(1+maxerror - 2*maxerror*rand())*d;(1+maxerror -
2*maxerror*rand())*w;(1+maxerror - 2*maxerror*rand())*w;(1+maxerror -
2*maxerror*rand())*d];
BR8=[(1+maxerror - 2*maxerror*rand())*d;(1+maxerror -
2*maxerror*rand())*w;(1+maxerror - 2*maxerror*rand())*w;(1+maxerror -
2*maxerror*rand())*d];
BR9=[(1+maxerror - 2*maxerror*rand())*d;(1+maxerror -
2*maxerror*rand())*w;(1+maxerror - 2*maxerror*rand())*w;(1+maxerror -
2*maxerror*rand())*d];
Battens(1)={BR1};Battens(2)={BR2};Battens(3)={BR3};Battens(4)={BR4};Bat
tens(5)={BR5};Battens(6)={BR6};Battens(7)={BR7};Battens(8)={BR8};Batten
s(9)={BR9};

D0=cell(9,1);
D0(1)={[[(1+maxerror - 2*maxerror*rand()),0;0,(1+maxerror -
2*maxerror*rand())]*D_1']'};...
    D0(2)={[[(1+maxerror - 2*maxerror*rand()),0;0,(1+maxerror -
2*maxerror*rand())]*D_2']'};...
    D0(3)={[[(1+maxerror - 2*maxerror*rand()),0;0,(1+maxerror -
2*maxerror*rand())]*D_3']'};...
    D0(4)={[[(1+maxerror - 2*maxerror*rand()),0;0,(1+maxerror -
2*maxerror*rand())]*D_4']'};...
    D0(5)={[[(1+maxerror - 2*maxerror*rand()),0;0,(1+maxerror -
2*maxerror*rand())]*D_5']'};...
    D0(6)={[[(1+maxerror - 2*maxerror*rand()),0;0,(1+maxerror -
2*maxerror*rand())]*D_6']'};...
    D0(7)={[[(1+maxerror - 2*maxerror*rand()),0;0,(1+maxerror -
2*maxerror*rand())]*D_7']'};...
    D0(8)={[[(1+maxerror - 2*maxerror*rand()),0;0,(1+maxerror -
2*maxerror*rand())]*D_8']'};

Pnts_int=[x(2),    y(2),     w/2;...
          xbot(2), ybot(2), w/2;...
          x(2),    y(2),     -w/2;...
          xbot(2), ybot(2), -w/2];

frontarm=cell(4,3,9);

for redsox=1:length(lbot);
    for celtics=1:4
        Pnts_int1=Pnts_int*[0,0,-1;0,1,0;1,0,0];
        frontarm{celtics,1,redsox}=Pnts_int1(celtics,1);
        frontarm{celtics,2,redsox}=Pnts_int1(celtics,2);
        frontarm{celtics,3,redsox}=Pnts_int1(celtics,3);
    end
```

```
[Pnts_outer,Residual]=BoxFind_3(Pnts_int,Longs{redsox},Battens{redsox},
D0{redsox});
    Pnts_int=Pnts_outer;
end
for celtics=1:4
    Pnts_outer1=Pnts_outer*[0,0,-1;0,1,0;1,0,0];
    frontarm{celtics,1,redsox+1}=Pnts_outer1(celtics,1);
    frontarm{celtics,2,redsox+1}=Pnts_outer1(celtics,2);
    frontarm{celtics,3,redsox+1}=Pnts_outer1(celtics,3);
end
frontarm=cell2mat(frontarm);
Pnts_all{4}=frontarm;
```

**BoxFind_3, calculates distal points given inner points of a truss bay:**

```
% Find box for truss, by Dr. Alan Jennings
function [Pnts_outer,Risidual]=BoxFind_3(Pnts_int,Longs,Battens,D0)

% Pnts_int [4,3] [{Top Front, Bottom Front, Top Back, Bottom
Back},{x,y,z}]
%               These are the four initial points for the box
%               Use the output of the function as the start for the
next box
%               X-out along the arm, Y-towards focus, Z-towards
%               observer/span-wise
% Longs [4,1] [Top Front; Bottom Front; Top Back; Bottom Back]
%               Lengths of the longerons,
%               tops are nominal, bottoms change to get desired
curvature
% Battens [4,1] [Front (short), bottom (long), top (long), back
(short)]
% D0 [2,1] [refernce length for a diagonal
% Pnts_all [4,3] [{Top Front, Bottom Front, Top Back, Bottom
Back},{x,y,z}]
%               These are the four outer points for the box
% Risidual [12,1] vector measuring the consistency of the constraints

%%
% intialize outer points
Pnts_outer=zeros(size(Pnts_int));

%% Use diagonal from bottom-front-inner to find Top-Front-outer
Const_Fun=@(TFo) [...
    norm([TFo,Pnts_int(1,3)]-Pnts_int(1,:))-Longs(1);...
    norm([TFo,Pnts_int(1,3)]-Pnts_int(2,:))-D0(1)];
[x,resnorm,residual,exitflag,output] = lsqnonlin(Const_Fun,...
    [Pnts_int(1,1)+Longs(1),Pnts_int(1,2)],...
    [],[],optimset('Algorithm','Levenberg-Marquardt','display','off'));
Pnts_outer(1,:)=[x,Pnts_int(1,3)];


%% repeat on bottom corner
Const_Fun=@(BBo) [...
    norm([BBo,Pnts_int(4,3)]-Pnts_int(4,:))-Longs(4);...
    norm([BBo,Pnts_int(4,3)]-Pnts_int(3,:))-D0(2)];
[x,resnorm,residual,exitflag,output] = lsqnonlin(Const_Fun,...
    [Pnts_int(4,1)+Longs(4),Pnts_int(4,2)],...
    [],[],optimset('Algorithm','Levenberg-Marquardt','display','off'));
Pnts_outer(4,:)=[x,Pnts_int(4,3)];

Const_Fun=@(BFo) [...
    norm(BFo-Pnts_int(2,:))-Longs(2);...
    norm(BFo-Pnts_outer(1,:))-Battens(1);...
    norm(BFo-Pnts_outer(4,:))-Battens(2)];

 [x,resnorm,residual,exitflag,output] = lsqnonlin(Const_Fun,...
```

```
    [mean(Pnts_outer([1,4],1)),Pnts_outer(4,2),Pnts_outer(1,3)],...
    [],[],optimset('Algorithm','Levenberg-Marquardt','display','off'));
Pnts_outer(2,:)=x;

Const_Fun=@(TBo) [...
    norm(TBo-Pnts_int(3,:))-Longs(3);...
    norm(TBo-Pnts_outer(1,:))-Battens(3);...
    norm(TBo-Pnts_outer(4,:))-Battens(4)];

[x,resnorm,residual,exitflag,output] = lsqnonlin(Const_Fun,...
    [mean(Pnts_outer([1,4],1)),Pnts_outer(1,2),Pnts_outer(4,3)],...
    [],[],optimset('Algorithm','Levenberg-Marquardt','display','off'));
Pnts_outer(3,:)=x;

%% show results

figure(23);%clf;
plot3(Pnts_int([1,2,4,3,1],3),Pnts_int([1,2,4,3,1],1),Pnts_int([1,2,4,3
,1],2),'bo-',...

Pnts_outer([1,2,4,3,1],3),Pnts_outer([1,2,4,3,1],1),Pnts_outer([1,2,4,3
,1],2),'ro-',...
      [Pnts_int(:,3),Pnts_outer(:,3),NaN+Pnts_outer(:,3)].',...
      [Pnts_int(:,1),Pnts_outer(:,1),NaN+Pnts_outer(:,1)].',...
      [Pnts_int(:,2),Pnts_outer(:,2),NaN+Pnts_outer(:,2)].','g+-',...
      [Pnts_int([2,3],3),Pnts_outer([1,4],3),NaN+[0;0]].',...
      [Pnts_int([2,3],1),Pnts_outer([1,4],1),NaN+[0;0]].',...
      [Pnts_int([2,3],2),Pnts_outer([1,4],2),NaN+[0;0]].','m--');
hold on
daspect([1,1,1]);grid on;
xlabel('z, span');ylabel('x, length');zlabel('y, focus');


Risidual=[...
    Longs(1)-norm(Pnts_int(1,:)-Pnts_outer(1,:));...
    Longs(2)-norm(Pnts_int(2,:)-Pnts_outer(2,:));...
    Longs(3)-norm(Pnts_int(3,:)-Pnts_outer(3,:));...
    Longs(4)-norm(Pnts_int(4,:)-Pnts_outer(4,:));...
    Battens(1)-norm(Pnts_outer(1,:)-Pnts_outer(2,:));...
    Battens(2)-norm(Pnts_outer(2,:)-Pnts_outer(4,:));...
    Battens(3)-norm(Pnts_outer(1,:)-Pnts_outer(3,:));...
    Battens(4)-norm(Pnts_outer(3,:)-Pnts_outer(4,:));...
    D0(1)-norm(Pnts_int(2,:)-Pnts_outer(1,:));...
    D0(2)-norm(Pnts_int(3,:)-Pnts_outer(4,:));...
    Pnts_outer(1,3)-Pnts_int(1,3);...
    Pnts_outer(4,3)-Pnts_int(4,3)];
```

**ToDATFile_2.m, builds NASTRAN cards, contains material properties:**

```
% takes points and makes NASTRAN code
function ToDATFile_2(Pnts_all,FileName)
% Pnts_all {4,1}[4,3,9] {arm}[(TF BF TB BB), (x,y,z), (Face: root to
tip)]
%               X-out along the arm, Y-towards focus, Z-towards
%               observer/span-wise
%               These are all the points for the arm, from these points
%               battens, longerons and diagionals are made
% FileName is the name for the text file where the results are saved

% V2, reorganized to match FEMAP order, helps for checking output

% Parameters
Alpha_T_Diag=2.34e-5; %MUST match value in the MAT card (approx line
300)

%% ordinary grid points
GridID_fun=@(Arm,Face,Point) num2str(Arm*1e3+Face*1e1+Point);

%% temputure loading function

Temp_Load_Fun=@(XYZ_1,XYZ_2,L_ideal,Alpha_T) (norm(XYZ_1-XYZ_2)-
L_ideal)/...
    (norm(XYZ_1-XYZ_2)*Alpha_T);
% NEED TO define for actual arms

%% element ID function

% this generates the unique element id's for each element type
ElemID_fun=@(Type,Arm,Face,Beam)
num2str(Type*1e4+Arm*1e3+Face*1e1+Beam);
% Type: 1-Longeron, 2-Short Battens, 3- Long Battens, 7- Diagonals
% Arm: 1- (+X) axis, 2- (+Z) axis,3- (-X) axis,4- (-Z) axis,
% Face: (1-root face battens only), 2- 1st box, ..., 9- Tip face
% Beam: Elements based on specific element type
%   for Longeron: 1-Top Front, 2-Bottom Front, 3-Top Back, 4-Bottom
Back
%   for short batten: 1-Front, 2-Back
%   for long batten: 1-Top, 2-Bottom
%   for diagonals: Previous face point given first, then outer face
point
%       1-TF to BF, 2-TB to BB, 3-BF to TF, 4-BB to TB, (outside faces)
%       5-TF to TB, 6-BF to BB, 7-TB to TF, 8-BB to BF, (upper/lower
faces)
%% This gives the ordering for the diagonals

Diag_PtNum=[1,2;3,4;2,1;4,3;1,3;2,4;3,1;4,2];
% row is diagonal number, column: 1- previous face, 2-outer face
% value is grid point index
```

```
%% open file for editing

copyfile('NASTRAN_Header_2c.blk',FileName)

fido=fopen(FileName,'at');
% use 'at' if you want to append, not throw out contents
% 'wt' for throwing out
%% write temputure cards

% default tempurature card
% $ Femap with NX Nastran Load Set 1 : Untitled
% TEMPD          1   71.45
Str_0=char(32*ones(10,8)); %formats the output string
% Str_0=repmat('12345678',10,1);
Str_0(1,1:5)='TEMPD'; %card name
Str_0(2,(end-1+1):end)='1'; % Load set number
Str_0(3,(end-4+1):end)='23.9'; % default tempurature
% write
fprintf(fido,'%8s',Str_0.');fprintf(fido,'\n');

% temperature loading cards
% $ Femap with NX Nastran Load Set 1 : Untitled
% TEMPRB         1   71031    173.    173.

Str_0=char(32*ones(10,8)); %formats the output string
% Str_0=repmat('12345678',10,1);
Str_0(1,1:6)='TEMPRB'; %card name
Str_0(2,(end-1+1):end)='1'; % Load set number
% row three is the element id where the load is applied
% rows four and five are the tempurtures at the first and second nodes
of
% the element (do the same)

% temp_PtNum=[1,2;3,4;2,1;4,3;1,3;2,4;3,1;4,2];
% % row is diagonal number, column: 1- previous face, 2-outer face
% % value is grid point index

Type=7; % for diagonals

temp_L_ideal_factor=[1.000;1.000; 1.000;1.000; 1.001;1.001;
0.999;0.999];
% Alpha_T_Diag=2.34e-5; %at top for input
for allie=1:length(Pnts_all)
    for barb=2:size(Pnts_all{allie},3)
        for carrie=1:size(Diag_PtNum,1)
Str=Str_0;
Str(3,(end-5+1):end)=ElemID_fun(Type,allie,barb  ,carrie);
XYZ_1=Pnts_all{allie}(Diag_PtNum(carrie,1),:,barb-1);
XYZ_2=Pnts_all{allie}(Diag_PtNum(carrie,2),:,barb  );
%
temp_y=num2str(Temp_Load_Fun(XYZ_1,XYZ_2,temp_L_ideal(carrie),temp_Alph
a_T),'%#8g');
temp_y=num2str(Temp_Load_Fun(XYZ_1,XYZ_2,...
```

```
    temp_L_ideal_factor(carrie)*norm(XYZ_1-
XYZ_2),Alpha_T_Diag),'%#8g');
temp_y=temp_y(1:min(length(temp_y),8));
Str(4,(end-length(temp_y)+1):end)=temp_y;
Str(5,(end-length(temp_y)+1):end)=temp_y;

% write
fprintf(fido,'%8s',Str.');fprintf(fido,'\n');
% fprintf('%8s',Str.');fprintf('\n');
        end
    end
end

%% Single point constraint cards

% Single point constraint cards
% $ Femap with NX Nastran Constraint Set 1 : Untitled
% SPC1           1  123456    1011
% SPC1           1  123456    1012
% SPC1           1  123456    1013
% SPC1           1  123456    1014
% ...

% Fixed points at base: X and -X faces
Str_0=char(32*ones(10,8)); %formats the output string
% Str_0=repmat('12345678',10,1);
Str_0(1,1:4)='SPC1'; %card name
Str_0(2,(end-1+1):end)='1'; % Constraint Set ID
% NOT unique amoung cards, all cards with this ID are used
Str_0(3,(end-5+1):end)='12345'; % Constraint directions (Z rotation
free)
% each number in the string is a flag
% 1,2,3 constraining displacements in x,y,z directions
% 4,5,6 constrainint rotation      in the x,y,z directions
Str=Str_0;
% now list the points:
for carrie=1:4
Str(carrie+3,(end-4+1):end)=GridID_fun(1,1,carrie);
end
fprintf(fido,'%8s',Str.');fprintf(fido,'\n');
Str=Str_0;
for carrie=1:4
Str(carrie+3,(end-4+1):end)=GridID_fun(3,1,carrie);
end
fprintf(fido,'%8s',Str.');fprintf(fido,'\n');

% Fixed points at base: Z and -Z faces
Str_0(3,(end-5+1):end)='12356'; % Constraint directions (X rotation
free)
Str=Str_0;
% now list the points:
for carrie=1:4
Str(carrie+3,(end-4+1):end)=GridID_fun(2,1,carrie);
end
```

```
fprintf(fido,'%8s',Str.');fprintf(fido,'\n');
Str=Str_0;
for carrie=1:4
Str(carrie+3,(end-4+1):end)=GridID_fun(4,1,carrie);
end
fprintf(fido,'%8s',Str.');fprintf(fido,'\n');

%% Property cards

% Property card
% $ Femap with NX Nastran Property 407 : Longeron
% $ Femap with NX Nastran PropShape 407 : 6,0,0.05,0.,0.,0.,0.,0.0006
% $ Femap with NX Nastran PropOrient 407 : 6,0,0.,1.,2.,3.,4.,-1.,0.,0.
% PBEAM         407     5591.8736-42.3141-72.3141-7      0.4.6278-7
0.+PR   BB
% +PR   BB      0.    -.05     .05      0.      0.     .05    -.05
0.+PA   BB
% +PA   BB    YESA      1.
+PC   BB
% +PC   BB.5306349.5306349

% longeron
Str_0=char(32*ones(10,8)); %formats the output string
% Str_0=repmat('12345678',10,1);
Str_0(1,1:5)='PBEAM'; %card name
Str_0(2,(end-3+1):end)='501'; % Element property ID number
Str_0(3,(end-3+1):end)='550'; % Material ID number
Str_0(4,(end-8+1):end)='1.8736-4'; % Area
Str_0(5,(end-8+1):end)='2.3141-7'; % Moment of inertia about 1st axis
Str_0(6,(end-8+1):end)='2.3141-7'; % Moment of inertia about 2nd axis
% these should be equal (for symmetric cross section), otherwise the
member
% cross section direction needs to be accounted for on the element
cards
Str_0(7,(end-2+1):end)='0.'; % Cross moment of inertia
% this should be zero, again, due to symetry, or element card need a
proper
% direction
Str_0(8,(end-8+1):end)='4.6278-7'; % Torsional stiffness
Str_0(9,(end-2+1):end)='0.'; % non-structural mass
fprintf(fido,'%8s',Str_0.');
fprintf(fido,'\n');
% Next line are the stress recovery points, not needed at this stage

% Short battens (Front and back, Top to Bottom)
Str_0=char(32*ones(10,8)); %formats the output string
% Str_0=repmat('12345678',10,1);
Str_0(1,1:5)='PBEAM'; %card name
Str_0(2,(end-3+1):end)='502'; % Element property ID number
Str_0(3,(end-3+1):end)='550'; % Material ID number
Str_0(4,(end-8+1):end)='1.8736-4'; % Area
Str_0(5,(end-8+1):end)='2.3141-7'; % Moment of inertia about 1st axis
Str_0(6,(end-8+1):end)='2.3141-7'; % Moment of inertia about 2nd axis
```

```
% these should be equal (for symmetric cross section), otherwise the
member
% cross section direction needs to be accounted for on the element
cards
Str_0(7,(end-2+1):end)='0.'; % Cross moment of inertia
% this should be zero, again, due to symetry, or element card need a
proper
% direction
Str_0(8,(end-8+1):end)='4.6278-7'; % Torsional stiffness
Str_0(9,(end-2+1):end)='0.'; % non-structural mass
fprintf(fido,'%8s',Str_0.');
fprintf(fido,'\n');
% Next line are the stress recovery points, not needed at this stage

% long battens (Top and Bottom, Front to back)
Str_0=char(32*ones(10,8)); %formats the output string
% Str_0=repmat('12345678',10,1);
Str_0(1,1:5)='PBEAM'; %card name
Str_0(2,(end-3+1):end)='503'; % Element property ID number
Str_0(3,(end-3+1):end)='550'; % Material ID number
Str_0(4,(end-8+1):end)='1.8736-4'; % Area
Str_0(5,(end-8+1):end)='2.3141-7'; % Moment of inertia about 1st axis
Str_0(6,(end-8+1):end)='2.3141-7'; % Moment of inertia about 2nd axis
% these should be equal (for symmetric cross section), otherwise the
member
% cross section direction needs to be accounted for on the element
cards
Str_0(7,(end-2+1):end)='0.'; % Cross moment of inertia
% this should be zero, again, due to symetry, or element card need a
proper
% direction
Str_0(8,(end-8+1):end)='4.6278-7'; % Torsional stiffness
Str_0(9,(end-2+1):end)='0.'; % non-structural mass
fprintf(fido,'%8s',Str_0.');
fprintf(fido,'\n');
% Next line are the stress recovery points, not needed at this stage

% diagonals
Str_0=char(32*ones(10,8)); %formats the output string
% Str_0=repmat('12345678',10,1);
Str_0(1,1:5)='PBEAM'; %card name
Str_0(2,(end-3+1):end)='701'; % Element property ID number
Str_0(3,(end-3+1):end)='750'; % Material ID number
Str_0(4,(end-8+1):end)='1.8736-4'; % Area
Str_0(5,(end-8+1):end)='2.3141-7'; % Moment of inertia about 1st axis
Str_0(6,(end-8+1):end)='2.3141-7'; % Moment of inertia about 2nd axis
% these should be equal (for symmetric cross section), otherwise the
member
% cross section direction needs to be accounted for on the element
cards
Str_0(7,(end-2+1):end)='0.'; % Cross moment of inertia
% this should be zero, again, due to symetry, or element card need a
proper
% direction
Str_0(8,(end-8+1):end)='4.6278-7'; % Torsional stiffness
```

```
Str_0(9,(end-2+1):end)='0.'; % non-structural mass
fprintf(fido,'%8s',Str_0.');
fprintf(fido,'\n');
% Next line are the stress recovery points, not needed at this stage

%% Material Cards
% material card
% MAT1         559 68947.6             .33 2.713-9  2.34-5    23.9
+MT   FJ
% +MT   FJ    55.2
% MAT4         559  .17922 962964. 2.713-9

% for longeron, short battens and long battens
Str_0=char(32*ones(10,8)); %formats the output string
% Str_0=repmat('12345678',10,1);
Str_0(1,1:4)='MAT1'; %card name
Str_0(2,(end-3+1):end)='550'; % Material ID number
Str_0(3,(end-7+1):end)='68947.6'; % Young's modulus
% Str_0(4,end)=' '; % Shear modulus, not needed with poisson's ratio
Str_0(5,(end-3+1):end)='.33'; % Poisson's ratio
Str_0(6,(end-7+1):end)='2.713-9'; % denisty
Str_0(7,(end-6+1):end)='2.34-5'; % thermal expansion coefficient
Str_0(8,(end-4+1):end)='23.9'; % reference tempurture
fprintf(fido,'%8s',Str_0.');
fprintf(fido,'\n');

% for diagonals
Str_0=char(32*ones(10,8)); %formats the output string
% Str_0=repmat('12345678',10,1);
Str_0(1,1:4)='MAT1'; %card name
Str_0(2,(end-3+1):end)='750'; % Material ID number
Str_0(3,(end-7+1):end)='68947.6'; % Young's modulus
% Str_0(4,end)=' '; % Shear modulus, not needed with poisson's ratio
Str_0(5,(end-3+1):end)='.33'; % Poisson's ratio
Str_0(6,(end-7+1):end)='2.713-9'; % denisty
Str_0(7,(end-6+1):end)='2.34-5'; % thermal expansion coefficient
Str_0(8,(end-4+1):end)='23.9'; % reference tempurture
fprintf(fido,'%8s',Str_0.');
fprintf(fido,'\n');
%%
%% write the grid cards

Str_0=char(32*ones(10,8)); %formats the output string
% Str_0=repmat('12345678',10,1);
Str_0(1,1:4)='GRID'; %card name
Str_0(3,end)='0'; % coordinate system
Str_0(7,end)='0';

for allie=1:length(Pnts_all)
    for barb=1:size(Pnts_all{allie},3)
% reset Str (e.g. if there's not as many digits in the number)
        for carrie=1:size(Pnts_all{allie},1)
Str=Str_0;
Str(2,(end-4+1):end)=GridID_fun(allie,barb,carrie);
```

```
temp_x=num2str(Pnts_all{allie}(carrie,1,barb),'%#8g');
temp_x=temp_x(1:min(length(temp_x),8));
%- sign adds a character, this still allows extra precision for
positive
%numbers, at the cost of shifting all negative numers to the positive
Str(4,(end-length(temp_x)+1):end)=temp_x;
temp_y=num2str(Pnts_all{allie}(carrie,2,barb),'%#8g');
temp_y=temp_y(1:min(length(temp_y),8));
Str(5,(end-length(temp_y)+1):end)=temp_y;
temp_z=num2str(Pnts_all{allie}(carrie,3,barb),'%#8g');
temp_z=temp_z(1:min(length(temp_z),8));
Str(6,(end-length(temp_z)+1):end)=temp_z;
% write
fprintf(fido,'%8s',Str.');
fprintf(fido,'\n');
        end
%         fprintf('%8s',Str.');
%         fprintf('\n');
    end
end

% reference point for beam orientations
Str=Str_0;
Str(2,(end-6+1):end)='800001';
Str(4,(end-2+1):end)= '0.';
Str(5,(end-3+1):end)='60.';
Str(6,(end-2+1):end)= '0.';
fprintf(fido,'%8s',Str.');
fprintf(fido,'\n');



%% write longeron cards
% the difference in the element-writing sections are which grid points
are
% connected

Str_0=char(32*ones(10,8)); %formats the output string
% Str_0=repmat('12345678',10,1);
Str_0(1,1:5)='CBEAM'; %card name
Str_0(3,(end-3+1):end)='501'; % property identification number, must
match property card
Str_0(6,(end-6+1):end)='800001'; % reference node for beam orientation
% since all cross sections are axially symetric, not really needed, but
% required to be defined. Point that is not colinear with any element.

Type=1; %longeron
for allie=1:length(Pnts_all)
    for barb=2:size(Pnts_all{allie},3)
        for carrie=1:size(Pnts_all{allie},1)
Str=Str_0;
Str(2,(end-5+1):end)=ElemID_fun(Type,allie,barb  ,carrie);
%   for Longeron: 1-Top Front, 2-Bottom Front, 3-Top Back, 4-Bottom
Back
```

```
Str(4,(end-4+1):end)=GridID_fun(     allie,barb-1,carrie);
Str(5,(end-4+1):end)=GridID_fun(     allie,barb-0,carrie);
% write
fprintf(fido,'%8s',Str.');fprintf(fido,'\n');
% fprintf('%8s',Str.');fprintf('\n');
        end
    end
end


%% write short batten cards

Str_0=char(32*ones(10,8)); %formats the output string
% Str_0=repmat('12345678',10,1);
Str_0(1,1:5)='CBEAM'; %card name
Str_0(3,(end-3+1):end)='502'; % property identification number, must
match property card
Str_0(6,(end-6+1):end)='800001'; % reference node for beam orientation
% since all cross sections are axially symetric, not really needed, but
% required to be defined. Point that is not colinear with any element.

Type=2; %short batten
for allie=1:length(Pnts_all)
    for barb=1:size(Pnts_all{allie},3)
        for carrie=1:2
Str=Str_0;
Str(2,(end-5+1):end)=ElemID_fun(Type,allie,barb  ,carrie);
%   for short batten: 1-Front (1-2), 2-Back (3-4)
Str(4,(end-4+1):end)=GridID_fun(     allie,barb  ,2*(carrie-1)+1);
Str(5,(end-4+1):end)=GridID_fun(     allie,barb  ,2*(carrie-1)+2);
% write
fprintf(fido,'%8s',Str.');fprintf(fido,'\n');
% fprintf('%8s',Str.');fprintf('\n');
        end
    end
end

%% write Long batten cards

Str_0=char(32*ones(10,8)); %formats the output string
% Str_0=repmat('12345678',10,1);
Str_0(1,1:5)='CBEAM'; %card name
Str_0(3,(end-3+1):end)='503'; % property identification number, must
match property card
Str_0(6,(end-6+1):end)='800001'; % reference node for beam orientation
% since all cross sections are axially symetric, not really needed, but
% required to be defined. Point that is not colinear with any element.

Type=3; %long batten
for allie=1:length(Pnts_all)
    for barb=1:size(Pnts_all{allie},3)
        for carrie=1:2
Str=Str_0;
Str(2,(end-5+1):end)=ElemID_fun(Type,allie,barb  ,carrie);
```

```
%   for long batten: 1-Top (1-3), 2-Bottom (2-4)
Str(4,(end-4+1):end)=GridID_fun(     allie,barb  ,carrie);
Str(5,(end-4+1):end)=GridID_fun(     allie,barb  ,carrie+2);
% write
fprintf(fido,'%8s',Str.');fprintf(fido,'\n');
% fprintf('%8s',Str.');fprintf('\n');
        end
    end
end

%% write diagonal cards

Str_0=char(32*ones(10,8)); %formats the output string
% Str_0=repmat('12345678',10,1);
Str_0(1,1:5)='CBEAM'; %card name
Str_0(3,(end-3+1):end)='701'; % property identification number, must
match property card
Str_0(6,(end-6+1):end)='800001'; % reference node for beam orientation
% since all cross sections are axially symetric, not really needed, but
% required to be defined. Point that is not colinear with any element.

Type=7; %diagonals
%   for diagonals: Previous face point given first, then outer face
point
%       1-TF to BF, 2-TB to BB, 3-BF to TF, 4-BB to TB, (outside faces)
%       5-TF to TB, 6-BF to BB, 7-TB to TF, 8-BB to BF, (upper/lower
faces)

for allie=1:length(Pnts_all)
    for barb=2:size(Pnts_all{allie},3)
        for carrie=1:size(Diag_PtNum,1)
Str=Str_0;
Str(2,(end-5+1):end)=ElemID_fun(Type,allie,barb  ,carrie);
Str(4,(end-4+1):end)=GridID_fun(     allie,barb-
1,Diag_PtNum(carrie,1));
Str(5,(end-4+1):end)=GridID_fun(     allie,barb
,Diag_PtNum(carrie,2));
% write
fprintf(fido,'%8s',Str.');fprintf(fido,'\n');
% fprintf('%8s',Str.');fprintf('\n');
        end
    end
end
%%
fprintf(fido,'ENDDATA\n');
%%
fclose(fido);
```

**Sigma_root_mean_square.m, calculates $\varepsilon_{RMS}$ from surface normal errors, corrects for ideal model inaccuracies:**

```
function [sigma_rms,sigma_rms_corrected]=sigma_root_mean_square(Data)

global ideal_norm_errors Data_Ideal

clc;
numel(Data_Ideal)
numel(ones(5))
numel(ideal_norm_errors)
k=0;
norm_lengths_per_node=zeros(length(Data),(length(Data)-1)/2);
for n=1:length(Data);
    k=k+1
    r=0;
    for m=1:length(Data{n})-1; %Matrix in each data cell ends in a
throwaway row
        if mod(m,2); %only pulls odd values, which form the top surface
            Node_Point=Data{n}(m,2:4);

[vector_error_normal,length_error_normal]=Surface_Normal_Determination(
Node_Point); %calls function that solves for normal length
            norm_lengths_per_node((m-r),n)=length_error_normal; %builds
matrix, each column is a set of normal surface errors, one for each
point in a trial
        else
            r=r+1; %this counter moves everything over in
norm_lengths_per_node
        end
    end
end
% norm_lengths_per_node
q=size(norm_lengths_per_node); %numbers of rows for # of samples (72),
# of cols for # of trials (usu. 30, set by 'NASTRAN_BuilkProcesses_2b'

% ideal_norm_errors=norm_lengths_per_node; %only use this line setting
values for ideal truss corrections
% corrected_norm_lengths_per_node=ideal_norm_errors; same as above
corrected_norm_lengths_per_node=norm_lengths_per_node-
ideal_norm_errors; %used for correcting out errors from "ideal truss"

sigma_rms=zeros(1,length(Data));
for n=1:q(2);
    sigma_rms(n)=sqrt(sum(norm_lengths_per_node(:,n).^2)/q(1)); %this
produces RMS error, each number is for one trial
end

sigma_rms_corrected=zeros(length(Data),1);
for n=1:q(2);

sigma_rms_corrected(n)=sqrt(sum(corrected_norm_lengths_per_node(:,n).^2
)/q(1)); %this produces RMS error, each number is for one trial
```

```
end

sigma_sort=sort(sigma_rms);
corrected_sigma_sort=sort(sigma_rms_corrected);

% figure1 = figure;
% axes1 = axes('Parent',figure1,'FontSize',20,'FontName','Times New
Roman');
% box(axes1,'on');
% hold(axes1,'all');

%
plot(sigma_sort*1000,(1:size(sigma_rms',1))/size(sigma_rms',1),'b','Lin
eWidth',2)
% % hold on
%
plot(corrected_sigma_sort*1000,(1:size(sigma_rms',1))/size(sigma_rms',1
),'r','LineWidth',2)
% ylabel('Percentage of Models','FontSize',24,'FontName','Times New
Roman');
% xlabel('Surface Normal Errors (mm)','FontSize',24,'FontName','Times
New Roman');
% % legend('Uncorrected RMS Surface Error','Corrected RMS Surface
Error');

sigma_rms_min=min(sigma_rms)*1000;
sigma_rms_mean=mean(sigma_rms)*1000;
sigma_rms_max=max(sigma_rms)*1000;

sigma_rms_corrected_min=min(sigma_rms_corrected)*1000
sigma_rms_corrected_mean=mean(sigma_rms_corrected)*1000
sigma_rms_corrected_max=max(sigma_rms_corrected)*1000

% sigma_sort_batten_vert=sort(sigma_rms);
% corrected_sigma_sort_all_10e5=sort(sigma_rms_corrected);
```

**Surface_Normal_Determination.m, calculates surface normal error, adds error in mesh connecting rods:**

```
function
[vector_error_normal,length_error_normal]=Surface_Normal_Determination(
Node_Point)

maxerror=-.0001; %value should be comparable to overall error
calculated
% maxerror=0; %zero error if calculating for ideal truss

x_e=Node_Point(1);
z_e=Node_Point(3);
y_e=Node_Point(2)+(1+maxerror - 2*maxerror*rand())*1.0541778; %adjusted
for error caused by truss offset from mesh
% y_e=Node_Point(2); %use this value for y if not accounting for truss
% offset of roughly 1 meter

% between center of truss and side of mesh
syms x y z

% Solving a Lagrangian by minimizing length between actual and ideal,
% constraint it must be on the surface
x_p=double(solve(x^2*(1+z_e^2/x_e^2)-320*160*x_e/x+320*(160-y_e),x)); %
ideal point
x_p=x_p(1); %ideal point
z_p=z_e*x_p(1)/x_e; %ideal point
y_p=(x_p(1)^2+z_p^2)/320; %ideal point

length_error_normal=norm([x_p-x_e;y_p-y_e;z_p-z_e]); %length of surface
error per node
vector_error_normal=[x_e-x_p,y_e-y_p,z_e-z_p];

% %% Plot error point vs 3d mesh w/ normal
% [xparafull,yparafull]=meshgrid(-75:2:75);
% zpos=(xparafull.^2+yparafull.^2)/320;
% surf(xparafull,yparafull,zpos) %mesh surface
% hold on
% plot3(x_e,z_e,y_e,'ro') %error point
% axis([-30 -10 -30 -10 -8 8])
%
% x_norm=-100:.5:100;
% y_norm=-100:.5:100;
% z_norm=-100:.5:100;
%
% X_norm=x_norm*(x_e-x_p)+x_e;
% Y_norm=y_norm*(y_e-y_p)+y_e;
% Z_norm=z_norm*(z_e-z_p)+z_e;
% plot3(X_norm,Z_norm,Y_norm)
%
% legend('surface','error point','error line')
```

**Mesh_tessellation_error.m, calculates surface normal error base on geometry of mesh panels vs. paraboloid**

```
%% check if errors are zero, essentially, 1E-15
bee=4.7:.1:3.071+4.7;
yellow=0:.1:3.37;

k=0;
Node_Points=cell(1,length(bee));
norm_lengths_per_mesh_node=zeros(length(bee),length(yellow));
for n=1:length(bee); %each cell corrsponds to 1 string of x-values
    k=k+1
    frog=zeros(1,length(yellow));
    Node_Points{n}=zeros(length(yellow),3);
    for m=1:length(yellow); %each row is constant x value, z values
differ over column
        frog(m)=(bee(n)^2+yellow(m)^2)/320;
        Node_Points{n}(m,:)=[bee(n), frog(m),yellow(m)];

[vector_error_normal,length_error_normal]=Surface_Normal_Determination(
Node_Points{n}(m,:));
        norm_lengths_per_mesh_node(n,m)=length_error_normal;
    end
end

epsilon_rms= sqrt(sum(norm_lengths_per_mesh_node.^2)/(n*m)); %this
produces mesh RMS error

%% for flat mesh rectangles, better RMS
bee=4.7:.1:3.071+4.7;
yellow=0:.11:3.37;

%plane
corners=[bee(1), (bee(1)^2+yellow(1)^2)/320, yellow(1);...
    bee(1), (bee(1)^2+yellow(end)^2)/320, yellow(end);...
    bee(end), (bee(end)^2+yellow(1)^2)/320, yellow(1);...
    bee(end), (bee(end)^2+yellow(end)^2)/320, yellow(end)];

vect_1=corners(4,:)-corners(1,:);
vect_2=corners(4,:)-corners(3,:);

vect_cross=cross(vect_1,vect_2);

k=0;
Node_Points=cell(1,length(bee));
norm_lengths_per_mesh_node_better=zeros(length(bee),length(yellow));
for n=1:length(bee); %each cell corrsponds to 1 string of x-values
    k=k+1
    frog=zeros(1,length(yellow));
    Node_Points{n}=zeros(length(yellow),3);
    for m=1:length(yellow); %each row is constant x value, z values
differ over column
```

```
        frog(m)=-(-vect_cross(1)*(bee(n)-corners(4,1))-
vect_cross(3)*(yellow(m)-corners(4,3)))/vect_cross(2)-corners(4,2);
%rise/run*x
        Node_Points{n}(m,:)=[bee(n), -frog(m)-.005,yellow(m)];

[vector_error_normal,length_error_normal]=Surface_Normal_Determination(
Node_Points{n}(m,:));
        norm_lengths_per_mesh_node_better(n,m)=length_error_normal;
    end
end

% epsilon_rms= sqrt(sum(sum(norm_lengths_per_mesh_node.^2))/(n*m))*1000
%this produces mesh RMS error in mm

%% for flat mesh rectangles, corners on the paraboloid
bee=4.7:.1:3.071+4.7;
yellow=0:.11:3.37;

%plane
corners=[bee(1), (bee(1)^2+yellow(1)^2)/320, yellow(1);...
    bee(1), (bee(1)^2+yellow(end)^2)/320, yellow(end);...
    bee(end), (bee(end)^2+yellow(1)^2)/320, yellow(1);...
    bee(end), (bee(end)^2+yellow(end)^2)/320, yellow(end)];

vect_1=corners(4,:)-corners(1,:);
vect_2=corners(4,:)-corners(3,:);

vect_cross=cross(vect_1,vect_2);

k=0;
Node_Points=cell(1,length(bee));
norm_lengths_per_mesh_node_corners=zeros(length(bee),length(yellow));
for n=1:length(bee); %each cell corrsponds to 1 string of x-values
    k=k+1
    frog=zeros(1,length(yellow));
    Node_Points{n}=zeros(length(yellow),3);
    for m=1:length(yellow); %each row is constant x value, z values
differ over column
        frog(m)=-(-vect_cross(1)*(bee(n)-corners(4,1))-
vect_cross(3)*(yellow(m)-corners(4,3)))/vect_cross(2)-corners(4,2);
%rise/run*x
        Node_Points{n}(m,:)=[bee(n), -frog(m), yellow(m)];

[vector_error_normal,length_error_normal]=Surface_Normal_Determination(
Node_Points{n}(m,:));
        norm_lengths_per_mesh_node_corners(n,m)=length_error_normal;
    end
end

% epsilon_rms= sqrt(sum(sum(norm_lengths_per_mesh_node.^2))/(n*m))*1000
%this produces mesh RMS error in mm
%% creating plots

radial_values=meshgrid(bee)';
```

```
z_values=meshgrid(yellow);

figure1 = figure;
axes1 = axes('Parent',figure1,'FontSize',20,'FontName','Times New
Roman');
box(axes1,'on');
hold(axes1,'all');

mesh(radial_values,
z_values,norm_lengths_per_mesh_node_corners*1000,ones(31)*.5)% errors
with corners on paraboloid
hold on
mesh(radial_values,
z_values,norm_lengths_per_mesh_node_better*1000,ones(31)) % errors with
center of mesh halfway to paraboloid

ylabel('Horizontal Distance Z (m)','FontSize',24,'FontName','Times New
Roman');
xlabel('Radial Distance X (m)','FontSize',24,'FontName','Times New
Roman');
zlabel('Normal Error Value (mm)','FontSize',24,'FontName','Times New
Roman')
legend('Panel Corners on Paraboloid','Panel Center Mid-way to Mesh');
```

## Bibliography


[1] J. R. Wertz, D. F. Everett and J. J. Puschell, Space Mission Engineering: The New SMAD, Hawthorne: Microcosm Press, 2011.

[2] NASA Goddard Space Flight Center, "Three Newly Designed Tracking and Data Relay Satellites To Help Replenish Existing On-Orbit Fleet," NASA, Greenbelt, 2001.

[3] D. Roddy, Satellite Communications, New York: McGraw Hill, 2006.

[4] NASA, "TDRS-L Media Kit," NASA Goddard, Greenbelt.

[5] G. Marks, C. Lillie and S. Kuehn, "Application of the AstroMech Reflector to Astrophysics Missions," Northrop Grumman Systems Corporation, 2011.

[6] Sea Launch, "Launch System: Rocket Segment Overview," [Online]. Available: http://www.sea-launch.com/launch-q11319-Rocket_Segment_Overview.aspx. [Accessed 31 January 2014].

[7] J. Black, "New Concepts for Very Large Antennas," AFIT, Dayton, 2012.

[8] J. M. Wilson, "The Design and Analysis of Electrically Large Custom-Shaped Reflector Antennas," AFIT, Wright-Patterson Air Force Base, 2013.

[9] J. M. Hedgepeth, "Accuracy Potentials for Large Space Antenna Reflectors with Passive Structure," *Journal of Spacecraft and Rockets,* vol. XIX, no. 3, pp. 211-217, 1982.

[10] A. Meguro, S. Harada and M. Watanabe, "Key technologies for high-accuracy large mesh antenna reflectors," *Acta Astronautica,* vol. 53, pp. 899-908, 2003.

[11] G. Greschik, M. M. Mikulas, R. G. Helms and R. E. Freeland, "Strip Antenna Figure Errors Due to Support Truss Member Length Imperfections," in *Structural Dynamics and Materials Conference,* Palm Springs, 2004.

[12] J. M. Hedgepeth, "Influence of Fabrication Tolerances on the Surface Accuracy of Large Antenna Structures," *AIAA Journal*, vol. XX, no. 5, pp. 680-686, 1981.

[13] Electronic Research Center, "Spacecraft Magnetic Torques," NASA, Washington D.C., 1969.

[14] Performance Composites Ltd, "Carbon Fibre Mechanical Properties," 2009. [Online]. Available: http://www.performance-composites.com/carbonfibre/mechanicalproperties_2.asp. [Accessed 5 February 2014].

[15] L. Meirovitch, Methods of Analytical Dynamics, Mineola: Dover Publications, 2003.

[16] R. J. Noll, "Zernike Polynomials and Atmospheric Turbulence," *Journal of the Optical Society of America*, vol. 66, pp. 207-211, 1976.

## Vita

Captain Jason C. Heller graduated as the Salutatorian from Frontier Regional High School in South Deerfield, Massachusetts. He entered undergraduate studies at the Worcester Polytechnic Institute in Worcester, Massachusetts where he graduated with a Bachelor of Science degree in Aerospace Engineering, with High Distinction, in May 2006. He was commissioned through the Detachment 340 AFROTC at the Worcester Polytechnic Institute where he was nominated for a Regular Commission.

His first assignment was at Yokota AFB, Japan as an officer in the 374th Civil Engineer Squadron. In July 2008, he was assigned to the 8th Civil Engineer Squadron, Kunsan AFB, Republic of Korea. In July 2009, he was assigned to the Electronic Systems Center staff at Hanscom AFB, Massachusetts. While stationed at Hanscom, he deployed overseas in July 2011 to Kandahar Air Field, Afghanistan for one year as the Regional Support Command-South Plans Officer in Charge. In August 2012, he entered the Graduate School of Engineering and Management, Air Force Institute of Technology. Upon graduation, he will be assigned to the 2nd Space Operations Squadron, Schriever AFB.

# REPORT DOCUMENTATION PAGE

*Form Approved*
*OMB No. 074-0188*



**PLEASE DO NOT RETURN YOUR FORM TO THE ABOVE ADDRESS.**

| 1. REPORT DATE *(DD-MM-YYYY)* | 2. REPORT TYPE | 3. DATES COVERED *(From – To)* |
|---|---|---|
| 27 Mar 2013 | Master's Thesis | Aug 2012–Mar 2014 |

**4. TITLE AND SUBTITLE**
FEASIBILITY OF VERY LARGE SPARSE APERTURE DEPLOYABLE ANTENNAS

**5a. CONTRACT NUMBER**

**5b. GRANT NUMBER**

**5c. PROGRAM ELEMENT NUMBER**

**6. AUTHOR(S)**
Heller, Jason C., Captain, USAF

**5d. PROJECT NUMBER**

**5e. TASK NUMBER**

**5f. WORK UNIT NUMBER**

**7. PERFORMING ORGANIZATION NAMES(S) AND ADDRESS(S)**
Air Force Institute of Technology
Graduate School of Engineering and Management (AFIT/ENY)
2950 Hobson Way, Building 640
WPAFB OH 45433-8865

**8. PERFORMING ORGANIZATION REPORT NUMBER**
AFIT-ENY-14-M-24

**9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES)**
Intentionally Left Blank

**10. SPONSOR/MONITOR'S ACRONYM(S)**

**11. SPONSOR/MONITOR'S REPORT NUMBER(S)**

**12. DISTRIBUTION/AVAILABILITY STATEMENT**
DISTRIBUTION STATEMENT A: APPROVED FOR PUBLIC RELEASE; DISTRIBUTION UNLIMITED

**13. SUPPLEMENTARY NOTES**
This material is declared work of the U.S. Government and is not subject to copyright protection in the United States.

**14. ABSTRACT**
The objective of this research is to explore the technical soundness of a very large, cross-shaped, parabolic, sparse aperture antenna extending 75 m from the bus. Specifically, describing the environment of the satellite, the effect of fabrication error on the structure and the remaining error budget for the system. The methodology involves creation of an ideal truss structure, to which all others are compared. A uniform distribution of proportional errors up to 1e-5 is introduced into the truss members' lengths and the models are subjected to a static Finite Element Analysis. A solution for the surface normal error is addressed using Lagrange multipliers. The goal is to hold the surface normal error for the entire satellite below a root mean square of 15 mm. The analysis yields a surface error of less than 1.53 mm, well within requirements. Despite the enormous size of the antenna reflector, and tight diameter/surface error ratio of 10,000 required for L-band communication, the system seems feasible. The values achieved for truss induced surface errors are in line with established techniques for analyzing full aperture, and strip, mesh antennas. With the mesh reflector and truss largely defined, nearly half of the 15 mm error budget remains.

**15. SUBJECT TERMS**
Large space structure, manufacturing deviation, communication satellite

| 16. SECURITY CLASSIFICATION OF: | | | 17. LIMITATION OF ABSTRACT | 18. NUMBER OF PAGES | 19a. NAME OF RESPONSIBLE PERSON |
|---|---|---|---|---|---|
| a. REPORT | b. ABSTRACT | c. THIS PAGE | | | Jennings, Alan L., PhD AFIT/ENY |
| U | U | U | UU | 103 | 19b. TELEPHONE NUMBER *(Include area code)* (937) 255-3636, x 7495 (alan.jennings@afit.edu) |